# MT90502

# Multi-Channel AAL2 SAR

Preliminary Datasheet

DS5420 ISSUE 1 November 2001

**Ordering Information**

MT90502AG 456 Pin Plastic BGA

**0 to +70°C**

## Features

- AAL2 Segmentation Reassembly device capable of simultaneously processing up to 1023 active CIDs (AAL2 Channel Identifier) and 1023 active VCs (Virtual Circuits).
- Support for up to 255 CIDs per VC. Maximum of 1023 CIDs.
- Implements AAL2 Common Part Sub-layer (CPS) functions specified in ITU I.363.2.
- Implements AAL2 Service Specific Convergence Sub-layer (SSCS) functions for G.711 PCM and G.726 ADPCM voice.
- Supports 44-byte PCM or ADPCM packet profiles specified in AF-VMOA-0145.00.
- CPS packet payload can support up to 64-bytes.
- Supports over-subscription of 10:1.
- H.100/H.110 compatible TDM bus for PCM or ADPCM data. Supports both master and slave TDM bus clock operation.
- TDM bus also supports compressed voice such as ITU G.723, G.728 and G.729 through HDLC encapsulation.
- Three UTOPIA Level 1 ports configurable as PHY or ATM allowing for connection to an external AAL5 SAR processor, or for chaining multiple MT90502 devices. Ports A & B are configurable as a single 8-bit UTOPIA Level 2 PHY port with 5 ADDR lines.
- UTOPIA module provides a cell switching function with a header translation.
- Performs silence suppression for PCM and ADPCM.
- Comfort noise generation.
- Capability to inject and recover CPS packets through the CPU host processor bus.
- 8-bit or 16-bit microprocessor port, configurable to Motorola or Intel timing.
- Single rail 3.3 V, 456 PBGA.
- IEEE 1149 (JTAG) interface.

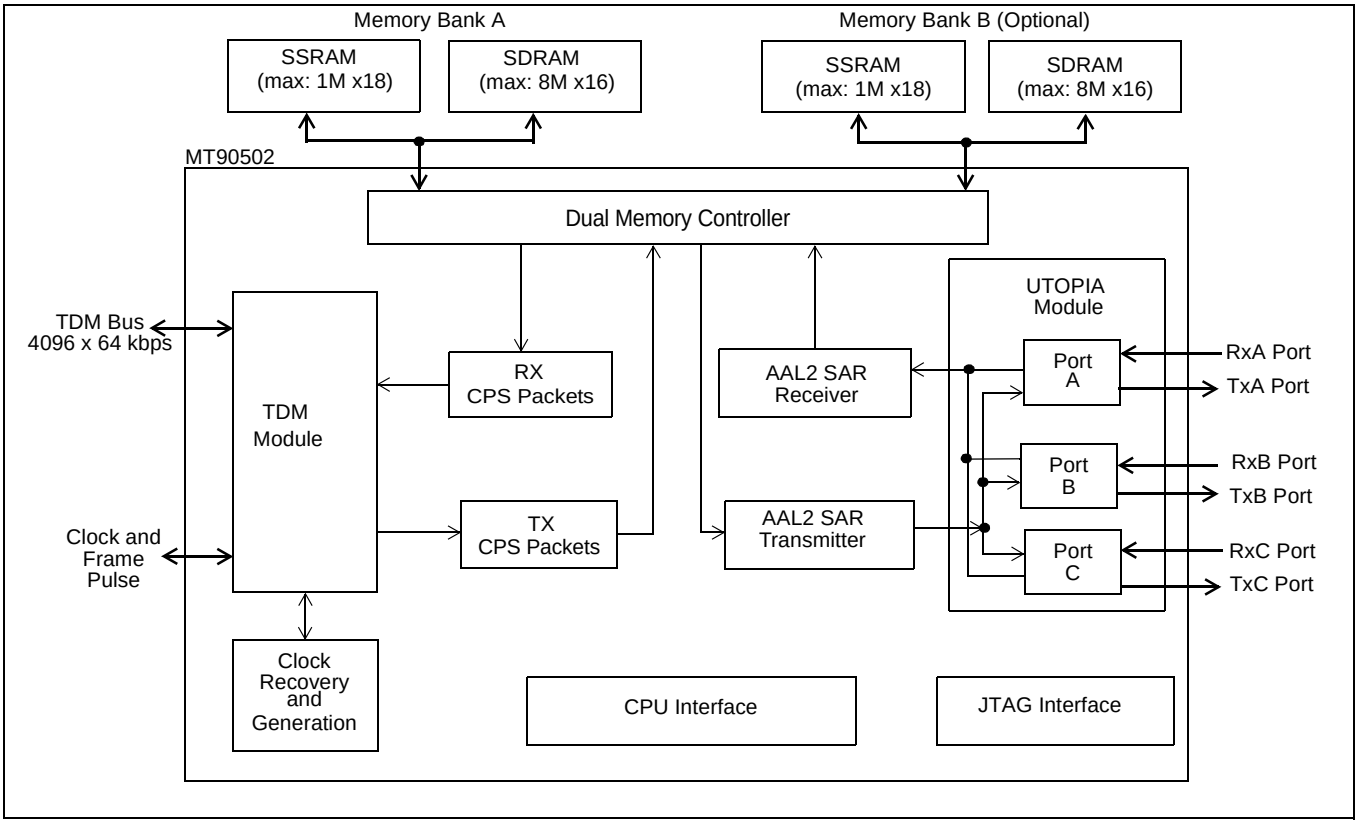



**Figure 1 - MT90502 Functional Block**

## Applications

- Gateway
- ATM Edge Switch
- Next Generation Digital Loop Carrier
- Multiservice Switching Platform
- 3rd Generation Mobile System Equipment

## Description

The MT90502 Multi-Channel AAL2 SAR bridges a standard TDM (Time Division Multiplexed) backplane to a standard ATM (Asynchronous Transfer Mode) bus. The device provides the CPS (Common Part Sublayer) and SAR (Segmentation and Reassembly) engines. The MT90502 has the capability of simultaneously processing 1023 bi-directional CIDs (AAL2 Channel Identifiers) and 1023 bi-directional VCs (Virtual Circuits). The device can be connected directly to an H.110 compatible bus. The TDM bus consists of 32 bi-directional serial data streams operating at 2.048, 4.096, or 8.192 Mbits/s.

The MT90502 directly accepts G.711 PCM (Pulse Code Modulation) and G.726 ADPCM (Adaptive Differential Pulse Code Modulation) traffic for packetisation. For these two data formats, the device also implements silence suppression and comfort noise generation. To support other voice compression algorithms, the MT90502 connects directly to commercially available DSPs through synchronous serial data streams. The Variable Bit Rate (VBR) traffic is HDLC encapsulated and carried over the serial data streams.

The interface to the ATM domain is provided by three UTOPIA Level 1 ports (Ports A, B, and C). All three of the UTOPIA ports can operate in ATM (master) or PHY (slave) mode. Ports A and B combined, architects a compliant UTOPIA Level 2 Multi-PHY port. The MT90502 provides the capability of routing ATM cells to different UTOPIA interfaces, SAR engine or CPU. This feature can be used to connect another MT90502 (to support up to 2046 CID channels or 2046 phone calls) and/or to connect an external AAL1 and/or AAL5 SAR.

**1.0 Pin-out ..... 7**

1.1 Pin Description Tables ..... 8

**2.0 Functional Description ..... 15**

2.1 CPU Interface ..... 15
2.1.1 CPU Interrupts ..... 15
2.1.1.1 Example Interrupt Flow ..... 15
2.1.1.1.1 Interrupt Initialisation ..... 16
2.1.1.1.2 Interrupt Servicing ..... 16
2.1.2 Intel/Motorola Interface ..... 16
2.1.2.1 Extended Indirect Accessing ..... 18
2.1.2.1.1 Extended Indirect Writes ..... 18
2.1.2.1.2 Extended Indirect Reads ..... 19
2.1.2.2 Extended Direct Accessing ..... 19
2.1.2.2.1 Extended Direct Writes ..... 19
2.1.2.2.2 Extended Direct Reads ..... 19
2.1.3 MT90502 Reset Procedure ..... 20
2.2 TDM Transmission ..... 21
2.2.1 Low-Latency Loopback Channels ..... 22
2.2.2 Treatment of PCM/ADPCM Data ..... 22
2.2.2.1 CPS-Packet Length ..... 24
2.2.2.2 TDM Data Formats ..... 24
2.2.2.3 Phase Alignment ..... 29
2.2.2.4 PCM/ADPCM CPS-Packet Assembly Structure ..... 31
2.2.3 Treatment of HDLC Data ..... 32
2.2.3.1 HDLC Streams ..... 32
2.2.3.2 Address Bytes ..... 32
2.2.3.3 Control Bytes and Length ..... 34
2.2.3.4 “Raw” AAL2 CPS-Packets ..... 34
2.2.4 CPS-Packet Final Assembly ..... 35
2.2.4.1 CPU CPS-Packets ..... 36
2.2.4.2 CPS-Packet Descriptor Queue ..... 38
2.2.4.3 TDM Frame Buffer ..... 39
2.3 TX SAR ..... 40
2.3.1 Overview ..... 40
2.3.2 AAL2 Cell Assembly Process ..... 40
2.3.2.1 AAL2 Cell Assembly Procedure ..... 42
2.3.3 AAL0 Cells ..... 43
2.4 RX SAR ..... 44
2.4.1 RX AAL2 VC Structure ..... 44
2.4.2 CID Structure ..... 45
2.4.3 CPS-Packet Disassembly Structures ..... 46
2.4.4 CPS-Packet Loss Compensation ..... 52
2.4.5 CPU CPS-Packets ..... 52
2.4.6 Treatment of Data Cells ..... 53
2.4.7 Errors and Events ..... 53
2.5 TDM Reception ..... 56
2.5.1 Overview ..... 56
2.5.2 RX Channel Association Memory ..... 57
2.5.3 RX Channel Underrun Condition ..... 57
2.5.4 Compression ..... 59
2.5.5 HDLC ..... 60
2.6 UTOPIA ..... 62
2.6.1 Overview ..... 62

2.6.2 UTOPIA Interfaces ..... 63
2.6.3 LED Operation ..... 63
2.6.4 Errors on Received Cells ..... 63
2.6.5 Cell Routing ..... 64
2.6.5.1 Mask & Match Process ..... 64
2.6.5.2 Look-Up Tables Entries ..... 65
2.6.5.3 LUT Addressing ..... 66
2.6.6 UTOPIA Clocks ..... 67
2.6.7 External Interface Signals ..... 69
2.6.8 UTOPIA Flow Control ..... 69
2.7 H.100/H.110 Interface ..... 70
2.7.1 Overview ..... 70
2.7.2 Bus Signaling ..... 70
2.7.3 H.100/H.110 Slave ..... 70
2.7.4 Operating as a Slave ..... 71
2.7.5 Operating as a Master ..... 71
2.7.6 H.100/H.110 Clock Selection Guide ..... 72
2.8 Clock Recovery ..... 75
2.8.1 Overview ..... 75
2.8.1.1 Adaptive Clock Recovery Modules ..... 75
2.8.1.1.1 adapx_ref clock generation ..... 75
2.8.1.2 Multiplexers ..... 75
2.8.2 Adaptive Clock Recovery Modules ..... 75
2.8.2.1 adapx_ref Clock Generation ..... 78
2.8.3 Multiplexers ..... 78
2.9 Silence Suppression ..... 83
2.9.1 Overview ..... 83
2.9.2 Simple Silent Suppression ..... 83
2.9.2.1 Silent Bit Indication ..... 83
2.9.2.2 Last Byte Indication ..... 83
2.9.2.3 Match and Mask Determines Silence ..... 83
2.9.3 Complex Silent Suppression ..... 84
2.9.3.1 Complex Silent Suppression Operation ..... 85
2.9.3.1.1 PCM Law Table ..... 85
2.9.3.1.2 DC Offset Calculation ..... 86
2.9.3.1.3 Signal Energy Calculation ..... 86
2.9.3.2 CPS-Packet Silence State ..... 87
2.9.3.2.1 Silence Suppression State Table ..... 87
2.9.3.2.2 SID Transmission Operation ..... 87
2.9.3.2.3 SID Reception Operation ..... 87
2.9.4 Voice/Silence Timer ..... 90
2.10 HDLC ..... 94
2.10.1 HDLC Overview ..... 94
2.10.2 HDLC Format ..... 94
2.10.3 HDLC Bit-Wise Format ..... 97
2.10.4 HDLC Byte-Wise Format ..... 97
2.11 Memory ..... 97
2.11.1 Memory Map ..... 97
2.11.2 Memory Structures ..... 98
2.11.3 Mem_Clk and Upclk ..... 103
2.11.4 Memory Controller ..... 104
2.11.4.1 Overview ..... 104
2.11.4.2 Functionality ..... 104
2.11.5 Initializing SSRAM and SDRAM ..... 104
2.11.6 Memory Configuration ..... 105

**3.0 Register List ..... 107**

3.1 CPU Register ..... 107
3.2 Main Registers ..... 110
3.3 TX Registers ..... 125
3.4 RX Registers ..... 126
3.5 TX TDM Registers ..... 131
3.6 UTOPIA Registers ..... 136
3.7 H.100/H.110 Registers ..... 152
3.8 Miscellaneous Registers ..... 166
3.9 RX TDM Registers ..... 169

**4.0 Electrical Specification ..... 171**

4.1 DC Characteristics ..... 171
4.2 AC Characteristics ..... 173
4.3 Intel/Motorola Interface ..... 173
4.3.1 UTOPIA Interface ..... 181
4.3.2 External Memory Interface ..... 181
4.3.3 H.100/H.110 Interface ..... 182

**5.0 Glossary of Terminology ..... 186**

5.1 Standard Terms and Abbreviations ..... 186
5.2 Terms specific to AAL2 ..... 187
5.3 Terms specific to this specification ..... 187
5.4 Register types ..... 187
5.5 Units and Conventions ..... 188

**6.0 Mechanical Drawing ..... 189**

## 1.0 Pin-out

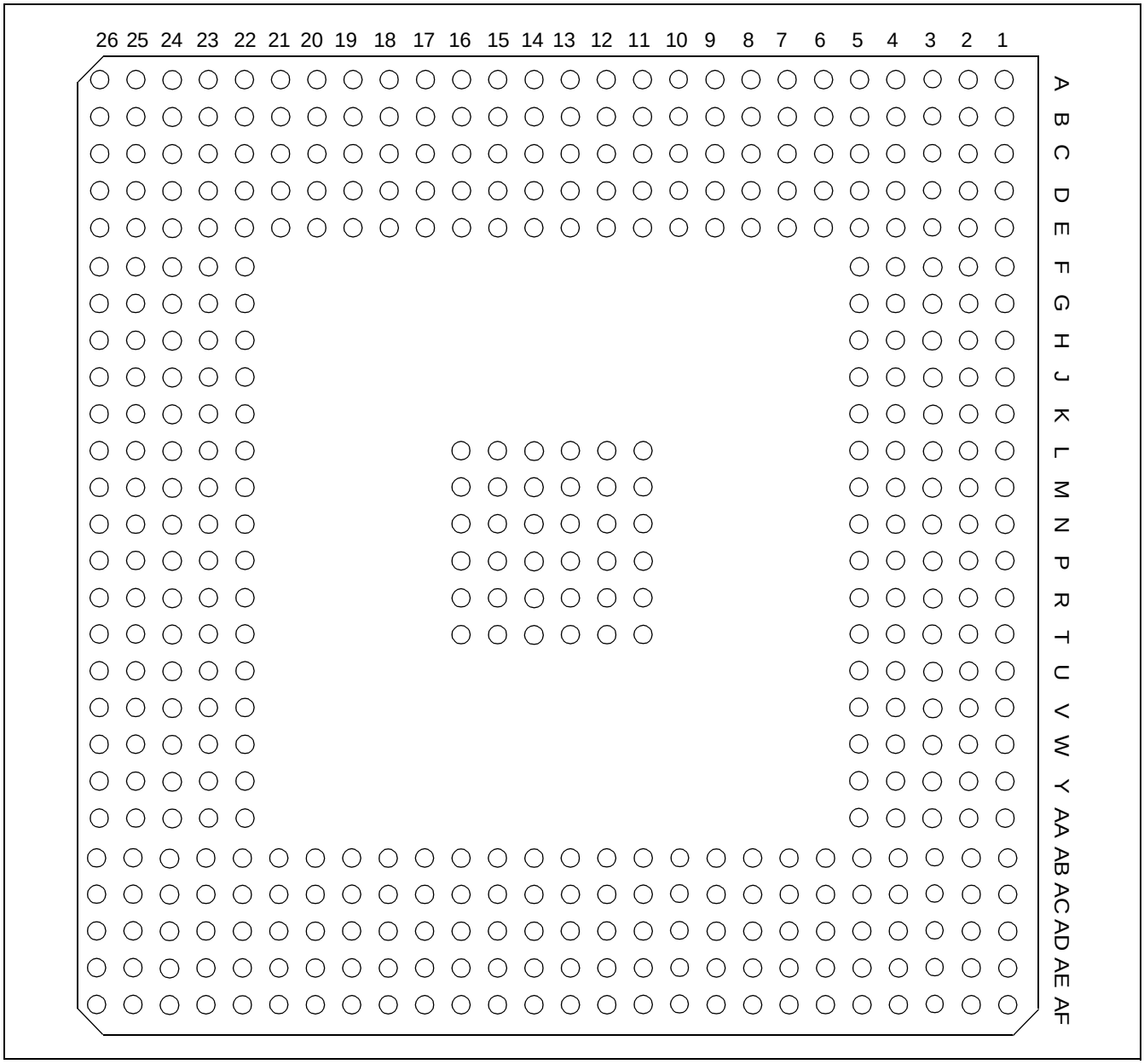



**Figure 2 - 456 PBGA**

The following tables contain each pin of the MT90502's main functional areas. A description of each pin is also provided.

Notes:

- All outputs are +3.3 $V_{DC}$.
- All input and output pins that are designated (F) can withstand 5 $V_{DC}$ being applied to them.
- All input and output pins that are designated (F) are tested with a 50 pF load unless otherwise specified.
- Designations under the "rst" (reset condition) table column are: X = undefined; Z = high impedance; 1 = high (+3.3 $V_{DC}$).
- I/O types include: Output (O), Input (I), Bidirectional (I/O), Power (PWR) and Ground (GND).
- All buses have pins listed in order from MSB to LSB.
- Pins with more than one function have their functions numbered.
- Unused H.100/H.110 input pins should be tied high with an external pull-up.

### 1.1 Pin Description Tables

| Pin | rst | Name | I/O | Type | Description |
|---|---|---|---|---|---|
| AC3 | | upclk | I | LVTTL (F) | CPU Clock. |
| F25, F23, J24, J23 | | cpu_mode[3:0] | I | LVTTL (F) | CPU Interface Mode Select (4 bits). |
| AA3, AA4, Y1, Y2, Y3, Y4, W1, W2, W3, W4, V1, V2, V3, V4, U1 | | cpu_a[14:0] | I | LVTTL (F) | CPU Address bus. |
| T2 | | cpu_$\overline{wr}$_r/$\overline{w}$ | I | LVTTL (F) | Intel Write or Motorola Read/Write |
| T1 | | cpu_$\overline{rd}$_$\overline{ds}$ | I | LVTTL (F) | Intel Read or Motorola Data Strobe |
| U4 | | cpu_ale | I | LVTTL (F) | Address Latch Enable |
| T4 | | cpu_a_das | I | LVTTL (F) | Direct Access Select. ‘1’ selects the direct address space. ‘0’ selects the indirection registers contained in the CPU interface. This pin can be connected to the MSB of an address bus. |
| U3 | | cpu_$\overline{cs}$ | I | LVTTL (F) | CPU chip select |
| T3, R1, R2, R3, R4, P1, P2, P3, P4, N1, N2, N3, N4, M1, M2, M3 | Z | cpu_d[15:0] | I/O | LVTTL 6 mA (F) | CPU Data bus |
| U2 | Z | cpu_rdy_$\overline{dtack}$ | I/O | LVTTL 6 mA (F) | Intel Ready or Motorola Data Ack. |
| AC2 | Z | interrupt1 | O | LVTTL 6 mA (F) | Interrupt 1 (configurable polarity) |
| AC1 | Z | interrupt2 | O | LVTTL 6 mA (F) | Interrupt 2 (configurable polarity) |

**Table 1 - CPU Bus Interface**

| Pins | rst | Name | I/O | Type | Description |
|---|---|---|---|---|---|
| C21 | Z | txa_led | I/O | LVTTL 12 mA (F) | UTOPIA port A TX LED |
| A19 | Z | rxa_led | I/O | LVTTL 12 mA (F) | UTOPIA port A RX LED |
| D21 | | rxa_alarm | I | LVTTL (F) | UTOPIA port A PHY alarm |
| D12 | Z | txa_clk | I/O | LVTTL 6 mA (F) | UTOPIA port A TX clock |
| D14 | Z | txa_soc | O | LVTTL 6 mA (F) | UTOPIA port A TX Start of Cell |
| A13 | Z | 1. $\overline{txa_enb}$<br>2. txa_clav | O | LVTTL 6 mA (F) | 1. UTOPIA port A TX Enable in ATM mode<br>2. UTOPIA port A TX Cell Available in PHY mode |
| B13 | | 1. txa_clav<br>2. $\overline{txa_enb}$ | I | LVTTL (F) | 1. UTOPIA port A TX Cell Available in ATM mode<br>2. UTOPIA port A TX Enable in PHY mode |
| C15, B15, A15, C14, B14, A14, D13, C13 | Z | txa_d[7:0] | O | LVTTL 6 mA (F) | UTOPIA port A TX Data bus |
| D15 | Z | txa_prty | O | LVTTL 6 mA (F) | UTOPIA port A TX Parity |
| A9 | Z | rxa_clk | I/O | LVTTL 6 mA (F) | UTOPIA port A RX clock |
| C12 | | rxa_soc | I | LVTTL (F) | UTOPIA port A RX Start of Cell |

**Table 2 - UTOPIA Interface Pins**

| Pins | rst | Name | I/O | Type | Description |
|---|---|---|---|---|---|
| B9 | Z | 1. $\overline{\text{rxa_enb}}$<br>2. rxa_clav | O | LVTTL 6 mA (F) | 1. UTOPIA port A RX Enable in ATM mode<br>2. UTOPIA port A RX Cell Available in PHY mode |
| C9 | | 1. rxa_clav<br>2. $\overline{\text{rxa_enb}}$ | I | LVTTL (F) | 1. UTOPIA port A RX Cell Available in ATM mode<br>2. UTOPIA port A RX Enable in PHY mode |
| A12, C11, B11, A11, D10, C10, B10, A10 | | rxa_d[7:0] | I | LVTTL (F) | UTOPIA port A RX Data bus |
| B12 | | rxa_prty | I | LVTTL (F) | UTOPIA port A RX Parity |
| D17 | Z | txb_led | I/O | LVTTL 12 mA (F) | UTOPIA port B TX LED |
| C17 | Z | rxb_led | I/O | LVTTL 12 mA (F) | UTOPIA port B RX LED |
| B17 | | rxb_alarm | I | LVTTL (F) | UTOPIA port B PHY alarm |
| D5 | Z | txb_clk | I/O | LVTTL 6 mA (F) | UTOPIA port B TX clock |
| D8 | Z | txb_soc | O | LVTTL 6 mA (F) | UTOPIA port B TX Start of Cell |
| A6 | Z | 1. $\overline{\text{txb_enb}}$<br>2. txb_clav | O | LVTTL 6 mA (F) | 1. UTOPIA port B TX Enable in ATM mode<br>2. UTOPIA port B TX Cell Available in PHY mode |
| B6 | | 1. txb_clav<br>2. $\overline{\text{txb_enb}}$ | I | LVTTL (F) | 1. UTOPIA port B TX Cell Available in ATM mode<br>2. UTOPIA port B TX Enable in PHY mode |
| B8, A8, D7, C7, B7, A7, D6, C6 | Z | txb_d[7:0] | O | LVTTL 6 mA (F) | UTOPIA port B TX Data bus |
| C8 | Z | txb_prty | O | LVTTL 6 mA (F) | UTOPIA port B TX Parity |
| E4 | Z | rxb_clk | I/O | LVTTL 6 mA (F) | UTOPIA port B RX clock |
| A4 | | 1. rxb_soc<br>2. txa_addr[4] | I | LVTTL (F) | 1. UTOPIA port B RX Start of Cell<br>2. txa_addr[4] when port A and B are combined |
| D3 | Z | 1. $\overline{\text{rxb_enb}}$<br>2. rxb_clav | O | LVTTL 6 mA (F) | 1. UTOPIA port B RX Enable in ATM mode<br>2. UTOPIA port B RX Cell Available in PHY mode |
| C1 | | 1. rxb_clav<br>2. $\overline{\text{rxb_enb}}$ | I | LVTTL (F) | 1. UTOPIA port B RX Cell Available in ATM mode<br>2. UTOPIA port B RX Enable in PHY mode |
| A2, D4, A1, B1, C2 | | 1. rxb_d[4:0]<br>2. rxa_addr[4:0] | I | LVTTL (F) | 1. UTOPIA port B RX Data bus [4:0]<br>2. rxa_addr[4:0] when port A and B are combined |
| A3, B3, B2 | | 1. rxb_d[7:5]<br>2. txa_addr[2:0] | I | LVTTL (F) | 1. UTOPIA port B RX Data bus [7:5]<br>2. txa_addr[2:0] when port A and B are combined |
| B4 | | 1. rxb_prty<br>2. txa_addr[3] | I | LVTTL (F) | 1. UTOPIA port B RX Parity.<br>2. txa_addr[3] when port A and B are combined. |

**Table 2 - UTOPIA Interface Pins (continued)**

| Pins | rst | Name | I/O | Type | Description |
|---|---|---|---|---|---|
| J4 | Z | txc_clk | I/O | LVTTL 6 mA (F) | UTOPIA port C TX clock |
| D2 | Z | txc_soc | O | LVTTL 6 mA (F) | UTOPIA port C TX Start of Cell |
| G1 | Z | 1. $\overline{\text{txc_enb}}$<br>2. txc_clav | O | LVTTL 6 mA (F) | 1. UTOPIA port C TX Enable in ATM mode<br>2. UTOPIA port C TX Cell Available in PHY mode |
| G2 | | 1. txc_clav<br>2. $\overline{\text{txc_enb}}$ | I | LVTTL (F) | 1. UTOPIA port C TX Cell Available in ATM mode<br>2. UTOPIA port C TX Enable in PHY mode |
| E3, E2, E1, F3, F2, F1, G4, G3 | Z | txc_d[7:0] | O | LVTTL 6 mA (F) | UTOPIA port C TX Data bus |
| D1 | Z | txc_prty | O | LVTTL 6 mA (F) | UTOPIA port C TX Parity |
| L1 | Z | rxc_clk | I/O | LVTTL 6 mA (F) | UTOPIA port C RX clock |
| H3 | | rxc_soc | I | LVTTL (F) | UTOPIA port C RX Start of Cell |
| L2 | Z | 1. $\overline{\text{rxc_enb}}$<br>2. rxc_clav | O | LVTTL 6 mA (F) | 1. UTOPIA port C RX Enable in ATM mode<br>2. UTOPIA port C RX Cell Available in PHY mode |
| L3 | | 1. rxc_clav<br>2. $\overline{\text{rxc_enb}}$ | I | LVTTL (F) | 1. UTOPIA port C RX Cell Available in ATM mode<br>2. UTOPIA port C RX Enable in PHY mode |
| H1, J3, J2, J1, K4, K3, K2, K1 | | rxc_d[7:0] | I | LVTTL (F) | UTOPIA port C RX Data bus |
| H2 | | rxc_prty | I | LVTTL (F) | UTOPIA port C RX Parity |

**Table 2 - UTOPIA Interface Pins (continued)**

| Pin | rst | Name | I/O | Type | Description |
|---|---|---|---|---|---|
| A22 | Z | ct_c8_a | I/O | LVTTL 24 mA (F) | H.100 8MHz clock A |
| A21 | Z | ct_c8_b | I/O | LVTTL 24 mA (F) | H.100 8MHz clock B |
| A24 | Z | ct_frame_a | I/O | LVTTL 24 mA (F) | H.100 Frame A |
| B22 | Z | ct_frame_b | I/O | LVTTL 24 mA (F) | H.100 Frame B |
| A17 | Z | ct_netref1 | I/O | LVTTL 24 mA (F) | H.100 Netref 1 |
| A16 | Z | ct_netref2 | I/O | LVTTL 24 mA (F) | H.100 Netref 2 |
| D18 | Z | ct_mc | I/O | LVTTL 24 mA (F) | H.100 Message Channel. If this pin is connected to the H100 bus, gpio[2] must be used to drive it. |
| C25, D25, E26, E25, E24, F26, F24, G26, G25, G24, G23, H24, J26, J25, K26, K24, L23, L24, M25, M24, N25, P24, P23, R24, R25, T26, U23, U26, V23, V24, V26, W23 | Z | ct_d[31:0] | I/O | PCI (F) | H.100 serial data bus |

**Table 3 - H.100/H.110 Interface Pins**

| Pin | rst | Name | I/O | Type | Description |
|---|---|---|---|---|---|
| B20 | Z | sclk | O | LVTTL 24 mA (F) | H.100 SCBUS system clock |
| C20 | Z | sclkx2 | O | LVTTL 24 mA (F) | H.100 SCBUS system clock x 2 |
| C18 | Z | c16p | O | LVTTL 24 mA (F) | H.100 H-MVIP 16 MHz clock positive output |
| A18 | Z | c16n | O | LVTTL 24 mA (F) | H.100 H-MVIP 16 MHz clock negative output |
| D20 | Z | c2 | O | LVTTL 24 mA (F) | H.100 MVIP-90 2 MHz clock |
| C19 | Z | c4 | O | LVTTL 24 mA (F) | H.100 MVIP-90 4 MHz clock |
| B21 | Z | frcomp | O | LVTTL 24 mA (F) | H.100 compatibility frame pulse |

**Table 3 - H.100/H.110 Interface Pins (continued)**

| Pin | rst | Name | I/O | Type | Description |
|---|---|---|---|---|---|
| AF2 | | mem_clk | I | LVTTL | memory clock (common to both bank A and B). It is also employed as the internal master clock. |
| AF11, AD10, AF10, AD9, AF9, AE8, AC7, AE7, AD7, AF8, AD8, AE9, AC9, AE10, AC10, AE11 | Z | mema_d[15:0] | I/O | LVTTL 6 mA | SDRAM/SSRAM bank A data bus |
| AD11, AF7 | Z | mema_p[1:0] | I/O | LVTTL 6 mA | SDRAM/SSRAM bank A parity bits |
| AF4, AE3 | 1 | mema_$\overline{cs}$[1:0] | O | LVTTL 6 mA | SSRAM bank A Chip selects 1, 0 |
| AE5 | Z | mema_r/$\overline{w}$ | O | LVTTL 6 mA | SSRAM bank A Read/Write |
| AE4, AF5 | Z | mema_$\overline{bws}$[1:0] | O | LVTTL 6 mA | SSRAM bank A Byte Write Selects 1:0 |
| AE12 | Z | mema_$\overline{cas}$ | O | LVTTL 6 mA | SDRAM bank A Column Address Select |
| AD12 | Z | mema_$\overline{ras}$ | O | LVTTL 6 mA | SDRAM bank A Row Address Select |
| AF12 | Z | mema_$\overline{we}$ | O | LVTTL 6 mA | SDRAM bank A Write Enable |
| AD6, AE6, AF6, AD5, AF3, AC12, AE13, AF13, AF14, AD13, AE14, AC14, AE15, AC15, AE16, AF16, AD15, AF15, AD14 | | mema_a[18:0] | O | LVTTL 6 mA | SDRAM/SSRAM bank A address bus |
| AE22, AD22, AF24, AE24, AD25, AC25, AB24, AB26, AB25, AC26, AD26, AE26, AE23, AF23, AD21, AE21 | Z | memb_d[15:0] | I/O | LVTTL 6 mA | SDRAM/SSRAM bank B data bus |
| AF22, AA24 | Z | memb_p[1:0] | I/O | LVTTL 6 mA | SDRAM/SSRAM bank B parity bits |
| W25, W26 | 1 | memb_$\overline{cs}$[1:0] | O | LVTTL 6 mA | SSRAM bank B Chip selects 1,0 |
| Y25 | Z | memb_r/$\overline{w}$ | O | LVTTL 6 mA | SSRAM bank B Read/Write |
| W24, Y26 | Z | memb_$\overline{bws}$[1:0] | O | LVTTL 6 mA | SSRAM bank B Byte Write Selects 1,0 |
| AC20 | Z | memb_$\overline{cas}$ | O | LVTTL 6 mA | SDRAM bank B Column Address Select |
| AD20 | Z | memb_$\overline{ras}$ | O | LVTTL 6 mA | SDRAM bank B Row Address Select |

**Table 4 - Memory Interface Pins**

| | | | | | |
|---|---|---|---|---|---|
| AF21 | Z | memb_$\overline{we}$ | O | LVTTL 6 mA | SDRAM bank B Write Enable |
| AA25, AA26, Y23, Y24, AC21, AE20, AC19, AF20, AE19, AD19, AF19, AE18, AC17, AE17, AD16, AF17, AD17, AF18, AD18 | | memb_a[18:0] | O | LVTTL 6 mA | SDRAM/SSRAM bank B address bus |

**Table 4 - Memory Interface Pins (continued)**

| Pin | rst | Name | I/O | Type | Description |
|---|---|---|---|---|---|
| AA1 | | reset | I | LVTTL (F) | Global Hardware Reset (active low) |
| AF26, AC24, AB23, AA23, T24, R26, N23, M23 | Z | gpio[7:0] | I/O | LVTTL 6 mA (F) | General Purpose I/Os |

**Table 5 - Miscellaneous Pins**

| Pin | Name | I/O | Type | Description |
|---|---|---|---|---|
| A26 | trst | I | LVTTL (F) | JTAG Test Reset |
| D22 | tck | I | LVTTL (F) | JTAG Test Clock |
| A25 | tdi | I | LVTTL (F) | JTAG Test Data In |
| C23 | tms | I | LVTTL (F) | JTAG Test Mode Select |
| A23 | tdo | O | LVTTL 6 mA (F) | JTAG Test Data Out |

**Table 6 - JTAG Pins**

| Pin | Name | I/O | Type | Description |
|---|---|---|---|---|
| AB2 | pll_clk | I | LVTTL (F) | PLL reference clock used for H.100 Master clock generation |
| A5 | PLLVDD1 | | | PLL Power Pin (3.3V). Place one .01uF, one 10uF and one 100pF capacitor near PLL_VDD1 / PLL_GND1 pins. |
| AB3 | PLLVDD2 | | | PLL Power Pin (3.3V). Place one .01uF, one 10uF and one 100pF capacitor near PLL_VDD2 / PLL_GND2 pins. |
| AE2 | PLLVDD3 | | | PLL Power Pin (3.3V). Place one .01uF, one 10uF and one 100pF capacitor near PLL_VDD3 / PLL_GND3 pins. |
| B5 | PLLGND1 | | | PLL Ground Pin (0V). |
| AB1 | PLLGND2 | | | PLL Ground Pin (0V). |
| AE1 | PLLGND3 | | | PLL Ground Pin (0V). |

**Table 7 - Phase Lock Loop (PLL) Pins**

VSS (0V): D9, D11, E5, E6, E9, E10, E13, E14, E17, E18, E21, E22, F4, F5, F22, H4, J5, J22, K5, K22, L4, L11, L12, L13, L14, L15, L16, M4, M11, M12, M13, M14, M15, M16, N5, N11, N12, N13, N14, N15, N16, N22, P5, P11, P12, P13, P14, P15, P16, P22, R11, R12, R13, R14, R15, R16, T11, T12, T13, T14, T15, T16, U5, U22, V5, V22, AA5, AA22, AB5, AB6, AB9, AB10, AB13, AB14, AB17, AB18, AB21, AB22, AF25.

**Table 8 - VSS (0V) Pins**

VDD3 (3.3 V): C3, C24, D23, E7, E8, E11, E12, E15, E16, E19, E20, G5, G22, H5, H22, L5, L22, M5, M22, R5, R22, T5, T22, W5, W22, Y5, Y22, AB7, AB8, AB11, AB12, AB15, AB16, AB19, AB20, AC4, AC23, AD3, AD24, AF1

**Table 9 - VDD3 (3.3V) Pins**

Note: If MT90502 is only connected to 3.3V devices on the H.100/H.110 bus, then 3.3V can be connected to the following pins. If any devices are 5V then these pins must be connected to 5V.

| VDD5 (3.3 V or 5.0 V): B25, D24, H26, L26, P25, U24 |
|---|

**Table 10 - VDD5 (3.3V or 5.0V) Pins**

| Not Connected (Leave Floating): A20, B16, B18, B19, B23, B24, B26, C4, C5, C16, C22, C26, D16, D19, D26, E23, H23, H25, K23, K25, L25, M26, N24, N26, P26, R23, T23, T25, U25, V25, AA2, AB4, AC5, AC6, AC8, AC11, AC13, AC16, AC18, AC22, AD1, AD2, AD4, AD23, AE25 |
|---|

**Table 11 - Not Connected (Leave Floating) Pins**

| Type | Input | Output | I/O | Power | Ground | N/C | Total |
|---|---|---|---|---|---|---|---|
| CPU Bus | 25 | 2 | 17 | | | | 44 |
| UTOPIA Port A | 13 | 12 | 4 | | | | 29 |
| UTOPIA Port B | 13 | 12 | 4 | | | | 29 |
| UTOPIA Port C | 12 | 12 | 2 | | | | 26 |
| H.100/H.110 | 0 | 7 | 39 | | | | 46 |
| Memory | 1 | 54 | 36 | | | | 91 |
| Miscellaneous | 1 | 0 | 8 | | | | 9 |
| JTAG | 4 | 1 | 0 | | | | 5 |
| PLL | 1 | 0 | 0 | 3 | 3 | | 7 |
| Power | | | | 46 | | | 46 |
| Ground | | | | | 79 | | 79 |
| No Connect | | | | | | 45 | 45 |
| Total: | 71 | 101 | 110 | 48 | 81 | 45 | 456 |

Note: Pins are listed under their main (default) function for UTOPIA ports A and B

**Table 12 - Pinout Summary**

# 2.0 Functional Description

## 2.1 CPU Interface

The MT90502 CPU module provides an interface permitting programmability from an external microprocessor or CPU. The CPU module permits read/write access to the MT90502's internal registers, internal memory and external memories.

The CPU interface is comprised of the following:

[1] Direct Access Select (DAS) as the MSB bit concatenated with a 15-bit address bus

[2] 16-bit data bus

[3] 2 interrupt signals

[4] associated control signals.

The CPU interface can be configured to operate in either Intel or Motorola mode. The MT90502 supports both 8-bit or 16-bit data bus and multiplexed or non-multiplexed address/data pins.

If the CPU is operating in 16-bit byte mode with the LSB of its address bus as a byte field, then the cpu_a[14:0] pins of the MT90502 can be connected to the a[15:1] pins of the CPU. If both the MT90502 and the CPU are in 16-bit word mode, then the cpu_a[14:0] pins should be connected to the a[14:0] pins of the CPU.

A reduced set of registers, the 'CPU Interface Registers' (000h to 00Ah), are employed to optimise access time and to permit the CPU to execute indirect read/write accesses. The CPU also engages these registers to perform direct read/write accesses. The MT90502 and CPU timing relationship is described in Section 4.3, "Intel/Motorola Interface," on page 173.

The CPU Control Register (100h) provides a software reset capability that allows the CPU to reset the MT90502 except for the CPU interface. The CPU interface can only be reset by a hardware reset.

### 2.1.1 CPU Interrupts

The CPU interface provides a programmable global interrupt capability. The interrupt signal names are 'interrupt1' and 'interrupt2', pins AC2 and AC1 respectively. Both interrupts have programmability to select their polarity (open collector drive) via registers 'interrupt1_conf' and 'interrupt2_conf', addresses 214h and 216h respectively. Interrupt1 provides the capability to program a minimum acceptable period between interrupts. The period is programmed in µs units via the 'interrupt1_conf' register. This provides a 'frequency interrupt controller' facility and masks the assertion of further interrupts until the specified period of time has elapsed. The mask period will start when the interrupt1_treated [15] bit in the register 'interrupt_flags' (address 210h) is set. When Interrupt2 is enabled it is always activated when an interrupt condition occurs.

The operation of the CPU interrupt network is common for all modules. When an interrupt is asserted, an interrupt flag is set to identify the module where the interrupt was generated. Each module has one or more Interrupt Enable Registers where a set interrupt enable bit enables an interrupt source. On completion of the ISR the interrupt must be cleared as the interrupt will remain asserted until it is de-asserted by the user. All Interrupt Enable Registers have a mirror Status Register. Hence, the bit positioning of the interrupt enables and the corresponding status bits are identical.

Note: Interrupt pins are always tri-stated when inactive.

#### 2.1.1.1 Example Interrupt Flow

Upon the initialisation of the Global Interrupt pins the following methodology is adopted to identify the source of the interrupt. For this example Interrupt2 is employed and the CPU module will be the source of the interrupt.

#### 2.1.1.1.1 Interrupt Initialisation

- Set interrupt polarity, register interrupt2_conf[15:14].
- Enable Interrupt2 for the CPU module by setting bit 0 in interrupt2_enable register (21Ah). The MT90502 will generate an interrupt on interrupt2 according to the modules enabled in interrupt2_enable.
- Set the individual CPU interrupt sources by enabling the respective bits in the ‘status0_ie’ register (104h). Within the ‘status0_ie’ register there are two possible interrupt sources: internal_read_timeout_ie and cpu_read_done_ie. In the MT90502 Register Description the interrupt bits are labelled IE (Interrupt Enable) in the ‘Type’ column. This register offers the facility to mask/disable unwanted interrupts.

#### 2.1.1.1.2 Interrupt Servicing

When interrupt2 is asserted (‘interrupt2’ pin):

- Read the interrupt flags to ascertain the module raising the interrupt. The CPU module interrupt flag is located in register interrupt_flags (210h), this bit is named cpureg_interrupt_active.
- If the cpureg_interrupt_active bit is set, locate the source of the CPU interrupt by reading the ‘status0’ at 102h, either internal_read_timeout and/or cpu_read_done.
- To de-assert the interrupt the user must write a 1 to register 102h bits 3 and/or 4, internal_read_timeout and cpu_read_done respectively. Only then will the interrupt be de-asserted.

### 2.1.2 Intel/Motorola Interface

| | |
|---|---|
| 000h | Control Register |
| 004h | Read/Write Data Register |
| 008h | Address High Register |
| 00Ah | Address Low Register |

The MT90502 CPU interface supports both Intel and Motorola modes with an 8-bit or 16-bit data bus and multiplexed or non-multiplexed address/data pins. The MT90502 supports 68MB of addressable space, therefore indirection addressing is necessary. The CPU interface directly addresses four control words, delegated for indirection accessing. The Indirection Register contents are shown in Table 14 to Table 17 inclusively. The timing relationship pertaining to the CPU Interface Registers and Extended Access is defined in Section 4.3 on page 173.

| cpu_mode [3:0] | Interface Type | ale | address pins | data pins | direct_access |
|---|---|---|---|---|---|
| 0000 | Intel, 16 bit data bus, non-multiplexed | cpu_ale* | cpu_a[14:0]** (word address) | cpu_d[15:0] | cpu_a_das |
| 0001 | Intel, 16 bit data bus, multiplexed | cpu_ale* | cpu_d[15:1]** (word address) | cpu_d[15:0] | cpu_a_das |
| 0010 | Intel, 8 bit data bus, non-multiplexed | cpu_ale* | cpu_a[14:0]** (byte address) | cpu_d[7:0] | cpu_a_das |
| 0011 | Intel, 8 bit data bus, multiplexed | cpu_ale* | cpu_a[14:8]** & cpu_d[7:0] (byte address) | cpu_d[7:0] | cpu_a_das |
| 0100 | Motorola, 16 bit data bus, non-multiplexed | cpu_ale* | cpu_a[14:0]** (word address) | cpu_d[15:0] | cpu_a_das |
| 0101 | Motorola, 16 bit data bus, multiplexed | cpu_ale* | cpu_d[15:1]** (word address) | cpu_d[15:0] | cpu_a_das |
| 0110 | Motorola, 8 bit data bus, non-multiplexed | cpu_ale* | cpu_a[14:0]** (byte address) | cpu_d[7:0] | cpu_a_das |
| 0111 | Motorola, 8 bit data bus, multiplexed | cpu_ale* | cpu_a[14:8]**& cpu_d[7:0] (byte address) | cpu_d[7:0] | cpu_a_das |
| 1xxx | Reserved | | | | |

* The cpu_ale pin is interpreted in all modes. However, it is not necessary in the non-multiplexed modes and can be tied to VCC.

* *The address placed on the cpu_a[14:0] pin is a word address in 16-bit mode and a byte address in 8-bit mode. The address, when placed on the cpu_d pins, is always a byte address.

**Table 13 - CPU Interface Mode Selection**

| Field | Bit | Type | Reset | Description |
|---|---|---|---|---|
| read_burst_length | 6:0 | RW | 01h | Number of words to prefetch: 00h = 128; 01h= 1; 02h = 2, etc. This field is set to 01h for individual (non-sequential) reads. |
| reserved | 7 | RO | 0h | Reserved. |
| access_req | 8 | PC | 0h | Set by software when an extended access is initialized. Reset by hardware when the access is completed. Used for extended indirect access only. |
| extended_a[3:1] | 11:9 | RW | 0h | Extended address bits 3:1. Invalid for extended direct access. |
| write_enable | 13:12 | RW | 0h | Active high write enables.<br>00 = read access.<br>01 = write to lower byte.<br>10 = write to upper byte.<br>11 = write to entire word.<br>This field is ignored for extended direct reads and all byte wide extended direct accesses. |
| extended_parity | 15:14 | RW | 0h | Read/Write Parity bits. |

**Table 14 - Control Register (000h)**

| Field | Bit | Type | Reset | Description |
|---|---|---|---|---|
| extended_data[15:0] | 15:0 | RW | 0000h | The extended indirect read/write data word register. Invalid for extended direct access. |

**Table 15 - Read/Write Data Register (004h)**

| Field | Bit | Type | Reset | Description |
|---|---|---|---|---|
| extended_a[32:20] | 12:0 | RW | 000h | Upper extended address [32:20]. |
| Reserved | 15:13 | RO | 0h | |

**Table 16 - Address High Register (008h)**

| Field | Bit | Type | Reset | Description |
|---|---|---|---|---|
| extended_a[19:4] | 15:0 | RW | 0000h | Lower extended address [19:4]. In extended direct addressing, bits 19:16 are employed for 16-bit data bus; bits 19:15 are employed or 8-bit data bus. |

**Table 17 - Address Low Register (00Ah)**

#### 2.1.2.1 Extended Indirect Accessing

Extended Indirect Accessing solely employees the registers 000h to 00Ah to access the 68MB of addressable memory space.

Synopsis: The access address is written to registers 000h, 008h, and 00Ah. The MT90502 will read/write to that address and fetch/place the data value from/to register 004h. For all extended indirect accesses the CPU_A_DAS bit will be held low.

##### 2.1.2.1.1 Extended Indirect Writes

The following steps must be executed to perform an extended indirect write:

1. Write the upper address, extended_a[32:20], to register 008h. This write may not be required if previous value holds true.
2. Write the lower address, extended_a[19:4], to register 00Ah. This write may not be required if previous value holds true.
3. Write the write data, extended_data[15:0], to register 004h. This write may not be required if previous value holds true.
4. Write write_enable, extended_parity, access_req=‘1’ and extended_a [3:1] in a single access to register 000h.
5. Read the access_req bit located in the Control Register[8] to determine when the write cycle has completed.

The software will set access_req [8] in register 000h (see Step 4 above). The hardware will reset the bit when the data write cycle has completed. Therefore, this bit can be polled to determine when the data write cycle has completed.

#### 2.1.2.1.2 Extended Indirect Reads

1. Write the upper address, extended_a[32:20], to register 008h. This write may not be required if previous value holds true.
2. Write the lower address, extended_a[19:4], to register 00Ah. This write may not be required if previous value holds true.
3. Write write_enable = 00, access_req='1' and extended_a [3:1] in a single access to register 000h.
4. Wait until access_req is cleared, then read the data from the data field extended_data[15:0], register 004h.

Optional parity check may be ascertained by performing a read on the extended_parity[15:14], register 000h. The software will set access_req[8] register 000h and the hardware will reset it when the data is ready to be read from register 004h.

### 2.1.2.2 Extended Direct Accessing

Extended Direct Accessing employs the high and low address registers to perform page addressing. The address within the page is provided directly by the CPU address bus. Similarly, the data is fetched/placed directly on the CPU data bus.

Synopsis: The access address is written to registers 008h and 00Ah. This will perform only the page addressing. Upon assertion of the address within the page, the MT90502 will read/write the data with respect to that address. The CPU_A_DAS bit is set when the data read/write occurs. When operating the CPU interface in direct mode with a 16-bit data bus, extended_a[19:16], are employed for the lower address word register 00Ah. However, when operating the CPU interface in direct mode with an 8-bit data bus, bits [19:15] are used for the lower address word.

#### 2.1.2.2.1 Extended Direct Writes

1. Write the upper address, extended_a[32:20], to register 008h. This write may not be required if previous value holds true.
2. Write the lower address extended_a[19:16] or [19:15] to register 00Ah. The remaining bits [15:4] or [14:4] are ignored. This write may not be required if previous value holds true.
3. Write write_enable[13:12] (this write may not be required if previous value holds true) and extended_parity[15:14]. The extended parity write is optional.
4. Write the data value to the address within the corresponding memory page with the CPU_A_DAS bit set.

#### 2.1.2.2.2 Extended Direct Reads

1. Write the upper address, extended_a[32:20], to register 008h. This write may not be required if previous value holds true.
2. Write the lower address, extended_a[19:16] or [19:15], to register 00Ah. The remaining bits [15:4] or [14:4] are ignored. This write may not be required if previous value holds true.
3. Assert the lower address within the memory page and fetch the read data with CPU_A_DAS set.
4. An optional read may be performed to obtain the parity values, extended_parity[15:14] register 000h.

### 2.1.3 MT90502 Reset Procedure

The following reset procedure is required to power-up the MT90502. The reset procedure must be adhered at power-up employing the nreset pin. All register accesses in the reset procedure may be performed in either Direct or Indirect mode.

1. Assert the nreset pin for approximately 1000 mem_clk periods.
2. De-assert the nreset pin.
3. Clear sreset bit in CPU Control Register (100h).
4. Configure mem_clk, upclk, and the led flashing frequencies in the CPU Registers (108h, 10Ah and 10Ch).
5. Configure the fast_clock PLL via registers 16Ch and 172h.
6. Configure the H.100/H.110 PLL in the CPU Register 174h.
7. Set active levels for interrupt pins in the Main Registers (214h and 216h).
8. Configure the UTOPIA port clocks in the Main Registers (220h, 222h, 224h, 228h, 22Ah and 22Ch).
9. Configure external memories in the Main Registers (230h, 232h, 234h, 240h and 242h).
10. Wait 10µs.
11. Set sreset bit in CPU Control Register (100h).
12. Initialise the SDRAM in the Main Registers (250h, 252h, 254h, 256h, 258h and 25Ah).

### 2.2 TDM Transmission

The TDM transmission module reads TDM data received by the H.100/H.110 interface, arranges the data in AAL2 CPS-Packets, and places the CPS-Packets in a FIFO for treatment by the TX SAR. The TX TDM module is capable of treating PCM data, ADPCM data and HDLC packets. Each channel can be configured as PCM, ADPCM or HDLC individually. For HDLC packets, zero-extraction (bit-wise or byte-wise - see Section 2.10 on page 94), and extraction of address, control and CRC bytes is performed. The received data is formatted into CPS-Packets that include CID, LI and UUI information.

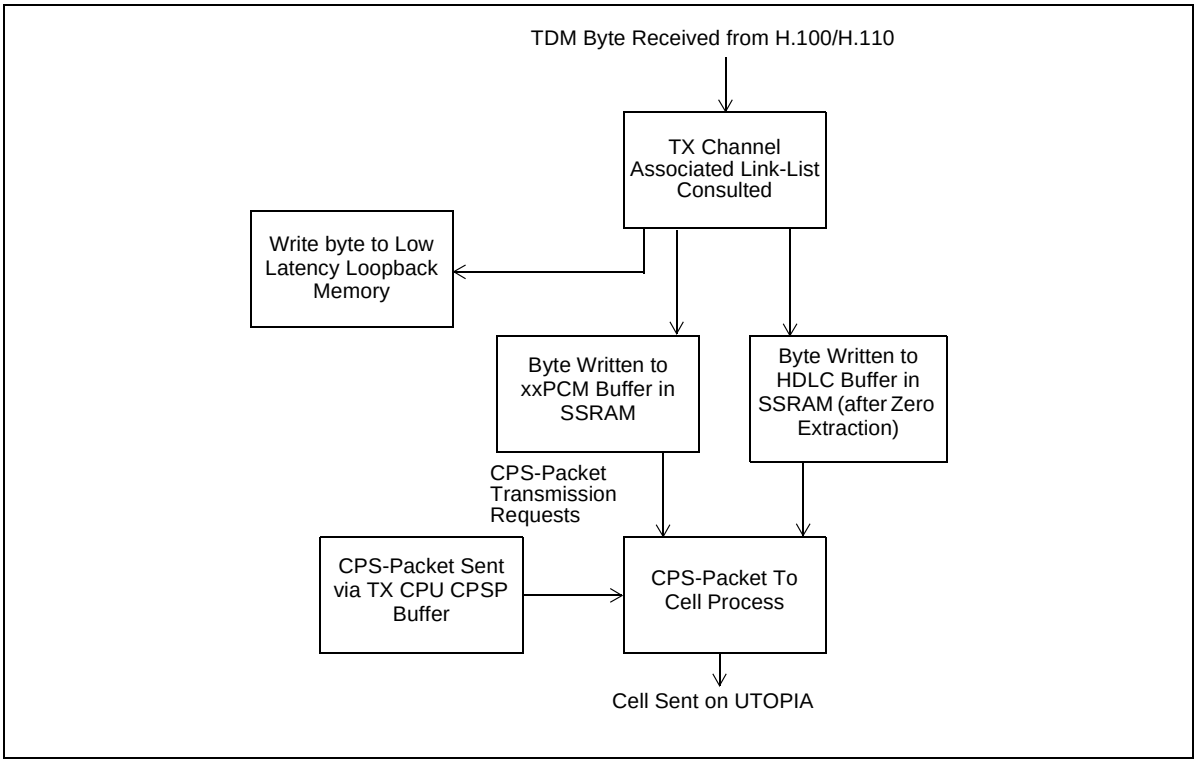



**Figure 3 - TX Cell Flow**

There are 32 streams on the H.100/H.110 bus, running at 2.048, 4.096 or 8.192MHz. Each individual time slot or DS0 channel on the H.100/H.110 bus is assigned a unique TSST (Time Slot/STream) number, according to the time slot number, stream number and the rate of the stream.

$$TSST=\begin{cases}\text{Time slot} * 32 + \text{Stream}, & \text{if Stream frequency is 8.192MHz;}\\ (\text{Time slot} * 2 + 1) * 32 + \text{Stream}, & \text{if Stream frequency is 4.096MHz;}\\ (\text{Time slot} * 4 + 3) * 32 + \text{Stream}, & \text{if Stream frequency is 2.048MHz.}\end{cases}$$

The time slot ranges are 0 to 31 for 2.048MHz streams, 0 to 63 for 4.096MHz stream, or 0 to 127 for 8.192MHz streams. Inside the MT90502 the TSST number is used to address any TDM channel position on the H.100/H.110 bus.

A TX TDM frame buffer exists in internal memory that contains the most recently received TDM bytes from each of the 4096 TSSTs. From the frame buffer, up to 1023-bytes are read each frame, according to the TSST link-list in TX Channel Association Memory (TX CAM). The TX CAM entry contains a channel number, an associated TSST, and a link to where the next valid entry exists in the TX CAM. The entries in the TX CAM **must** be linked sequentially according to TSST but can exist in any order in memory.

Each entry in TX CAM is one of the following four types:

1. The TSST in that entry carries PCM or ADPCM data. An xxPCM Channel Number should be assigned.
2. The TSST in that entry belongs to an HDLC stream. An HDLC Stream Number should be assigned.

3. The TSST in that entry is a loopback channel. A Low-Latency Loopback (LLL) Channel Number should be assigned.

4. The TSST in that entry is a phasing channel. The device will use the data on that TSST to synchronize its internal phase/sub-phase operation.

Each TSST that carries PCM, ADPCM or HDLC data needs a CPS-Packet assembly structure in TX TDM Control memory to have CPS-Packets assembled. The xxPCM channel number and HDLC Stream Number are used to indicate the memory location of the CPS-Packet assembly structure and the CPS-Packet final assembly structure.

### 2.2.1 Low-Latency Loopback Channels

A low-latency loopback (LLL) is a connection between any TX TDM channel and any RX TDM channel. Up to 128 loopback connections can be established.

Channels designated as LLL channels in the TX CAM are written directly into the appropriate location in the LLL memory of the RX TDM module. Once there, they are routed out to TSSTs that are assigned the same LLL channel numbers in RX CAM.

### 2.2.2 Treatment of PCM/ADPCM Data

PCM and ADPCM data is accumulated in a CPS-Packet circular buffer in TX SSRAM until the number of bytes pending equals the length of the AAL2 CPS-Packet to be assembled, as specified in the number of EDUs (# EDU) field of the PCM/ADPCM CPS-Packet assembly structure for that xxPCM channel. The TX TDM is capable of treating PCM at 64 kbps and compressed ADPCM at 40, 32, 24, or 16 kbps. For the compressed ADPCM channels, instead of capturing all 8-bits in an incoming TDM byte, the MT90502 will only sample 5-, 4-, 3-, or 2-bits of the byte, according to the data rate. The MT90502 can be configured to sample either the MSBs or the LSBs of TDM bytes; the configuration applies to all compressed channels.

| | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| +0 | | | | | | | | | | | | | | | | |
| +2 | | | | | | 0 | HDLC Stream/xxPCM Channel Number | | | | | | | | | |
| +4 | | | | TSST[11:0] | | | | | | | | | | | | |
| +6 | | | | | | | Link to next entry | | | | | | | | | |

HDLC Stream or xxPCM Channel Structure

| | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| +0 | | | | | | | | | | | | | | | | |
| +2 | | | | | | 1 | 0 | 00LL | 0 | LLL Channel Number | | | | | | |
| +4 | | | | TSST[11:0] | | | | | | | | | | | | |
| +6 | | | | | | | Link to next entry | | | | | | | | | |

LLL Channel Structure

| | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| +0 | | | | | | | | | | | | | | | | |
| +2 | | | | | | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | XPI | |
| +4 | | | | TSST[11:0] | | | | | | | | | | | | |
| +6 | | | | | | | Link to next entry | | | | | | | | | |

Phasing Channel Structure

Reserved

**TX CAM Notes:**

- The TX CAM consists of 1024 8-byte structures in internal memory at addresses 4000h to 5FF8h.
- One TX CAM entry exists for each selected TSST.
- The HDLC Stream/xx PCM Channel Number is used to locate the CPS-Packet assembly structure, the CPS-Packet final assembly structure (see Figures 12, 13, 14, 15 and 16) and the CPS-Packet buffer for the associated TSST.
- **Link to next entry:** A pointer to the next entry in the linked list of TX CAM entries. 000h indicates the end of the linked list (return to start). Entry 0 in the linked list is always examined at the start of frame and is never associated to a valid TDM channel.
- The TX CAM entries are each consulted once every frame.
- TX CAM entries can exist in any order in memory but must be linked in order of TSST.
- **XPI:** External Phase Indicator: “00” = 24 frame phase; “01” = 80 frame phase; “10” = 64 frame phase (also used for 8, 16 and 32 frame phase); “11” = 88 frame phase

**Figure 4 - TX Channel Association Memory (TX CAM)**

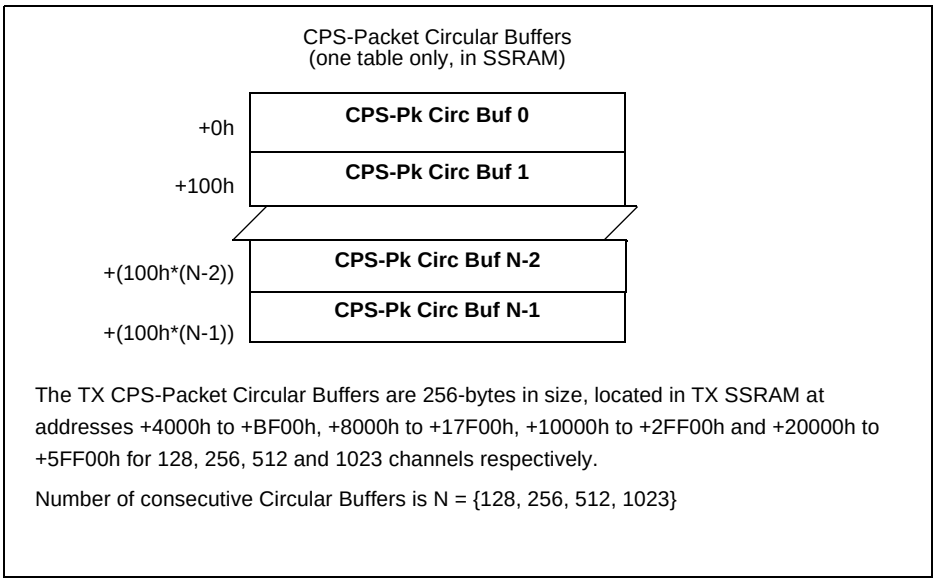



**Figure 5 - TX CPS-Packet Circular Buffers**

#### 2.2.2.1 CPS-Packet Length

An Encoding Data Unit (EDU) consists of 8 PCM or ADPCM data blocks. A data block is 8-bits for uncompressed PCM and 5-, 4-, 3-, or 2-bits for ADPCM compression rates of 40, 32, 24, or 16 kbps, respectively. AAL2 CPS-Packets may contain 1-, 2-, 3-, 4-, 5-, or 8-EDUs. The MT90502 also supports 44-byte PCM, 44-byte 32K ADPCM and 40-byte 32K-ADPCM CPS-Packets as per ATM Forum requirement for Loop Emulation Service. 40/80 frame PCM/ADPCM represents 40-byte 32K ADPCM, and 44/88 frame PCM/ADPCM represents both 44-byte PCM and 44-byte 32K ADPCM.

#### 2.2.2.2 TDM Data Formats

Two different formats are available for transferring PCM/ADPCM samples on the H.100/H.110 bus. The first format, Format A, uses a single time slot on the H.100/H.110 bus to transfer a sample. This format does not allow distinguishing between PCM and ADPCM samples and therefore does not allow automatic switching between PCM and ADPCM samples.

If a channel is configured as ADPCM, its compression rate can be determined on a byte by byte basis. This is done by using a certain byte format (see Figure 8) on the TDM bus in which the nibble is placed in either the high bits or the low bits of the byte, followed by a '1' and padding the rest of the byte with '0's. By so doing, the compression rate of the sample is coded implicitly into the byte. As a CPS-Packet can contain only one encoding scheme or compression rate, the compression rate may change only on CPS-Packet boundaries. Dynamic compression can be performed between all ADPCM compression rates.

The second format, Format B, uses two time slots on the H.100/H.110 bus to transfer a single PCM/ADPCM sample. Using two time slots will provide explicit indication of PCM or ADPCM sample in both the Tx and Rx direction. This will allow for dynamic switching between PCM and ADPCM encoding schemes.

Both Format A and B allow the MT90502 to send a reset signal to the external decompressor to indicate the end of a silent period. For silence suppression to function properly, both the ADPCM compressor and decompressor must be synchronously reset in between continuous talk spurts. This reset must take place before each non-silent packet that was preceded by a silent packet, given that silence suppression is enabled. The external decompressor must reset itself when it sees either b7 or b0, depending on the placement of ADPCM nibbles, is set in ADPCM samples. For Format B, a reset signal (PCM-R=11) will also be given when a voice (non-padding) sample is received that was preceded by a padding sample.

The 32 TDM streams on the H.100/H.110 bus are grouped into 8 quads (stream[3:0] is quad 0, stream[7:4] is quad 1, etc.), and the choice of data transfer format can be made on each individual quad using bits 13:6 in register 700h.

Figure 6 to Figure 11 show the different data formats for PCM/ADPCM samples for each of the different compression rates, encoding schemes and transfer modes.

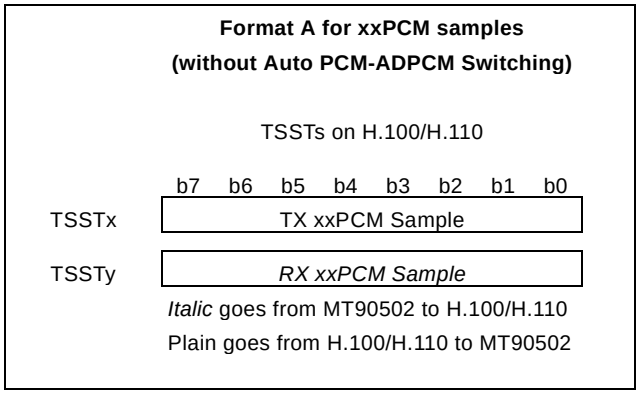



**Figure 6 - PCM/ADPCM Data Format A**

**Format A for RX xxPCM Samples**
**(without Auto PCM-ADPCM Switching)**

**Format of ADPCM Samples in low bits**
**(with Decompressor reset)**

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | ADPCM 40 kbps | | | | |
| 1 | 0 | 0 | 1 | 32 kbps | | | |
| 1 | 0 | 0 | 0 | 1 | 24 kbps | | |
| 1 | 0 | 0 | 0 | 0 | 1 | 16 kbps | |

**Format of ADPCM Samples in low bits**
**(no Decompressor reset)**

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | ADPCM 40 kbps | | | | |
| 0 | 0 | 0 | 1 | 32 kbps | | | |
| 0 | 0 | 0 | 0 | 1 | 24 kbps | | |
| 0 | 0 | 0 | 0 | 0 | 1 | 16 kbps | |

**Format of ADCM Samples in high bits**
**(with Decompressor reset)**

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| ADPCM 40 kbps | | | | 1 | 0 | 1 | |
| 32 kbps | | | | 1 | 0 | 0 | 1 |
| 24 kbps | | | 1 | 0 | 0 | 0 | 1 |
| 16 kbps | 1 | 0 | 0 | 0 | 0 | 1 | |

**Format of ADPCM Samples in high bits**
**(no Decompressor reset)**

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| ADPCM 40 kbps | | | | 1 | 0 | 0 | |
| 32 kbps | | | | 1 | 0 | 0 | 0 |
| 24 kbps | | | 1 | 0 | 0 | 0 | 0 |
| 16 kbps | 1 | 0 | 0 | 0 | 0 | 0 | |

**Format of PCM Padding Samples in ADPCM only time slots**

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| PCM[7] | 1 | PCM[5:0] | | | | | |

**Format of PCM samples**

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| PCM | | | | | | | |

It is recommended that b6 of PCM padding bytes be 1 - this is in-line with the requirement that the padding buffers contain all low energy samples, u-law max value of 471 out of 8159 and A-law max value of 252 out of 4096 - and that ADPCM samples be placed in the low bits. This will allow the decompressor to differentiate PCM padding bytes from ADPCM samples, in order to bypass the decompressor during a silence period.

Note that switching between ADPCM and PCM requires software intervention when using Format A

**Figure 7 - PCM/ADPCM RX Data Format A**

**Format A for TX xxPCM Samples**
**(without Auto PCM-ADPCM Switching)**

**Format of ADPCM Samples in high bits**

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| ADPCM 40 kbps | | | | | 1 | 0 | 0 |
| 32 kbps | | | | 1 | 0 | 0 | 0 |
| 24 kbps | | | 1 | 0 | 0 | 0 | 0 |
| 16 kbps | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

**Format of ADPCM Samples in low bits**

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | ADPCM 40 kbps | | | | |
| 0 | 0 | 0 | 1 | 32 kbps | | | |
| 0 | 0 | 0 | 0 | 1 | 24 kbps | | |
| 0 | 0 | 0 | 0 | 0 | 1 | 16 kbps | |

**Format of PCM Sample**

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| PCM | | | | | | | |

Note: switching between ADPCM and PCM requires software intervention in Format A.

**Figure 8 - PCM/ADPCM TX Data Format A**

**MT90502 Format B for xx PCM**
**(with Auto PCM-ADPCM Switching)**

TSST Grouping on H.100/H.110

| | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|
| TSST n | PCM | TX PCM Unsigned Magnitude | | | | | | |
| TSST n+1 | TX xxPCM Sample | | | | | | | |
| TSST n+2 | *PCM-R* | *Reserved* | | | | | | |
| TSST n+3 | *RX xxPCM Sample* | | | | | | | |

*Italic* goes from MT90502 to H.100/H.110

Plain goes form H.100/H.110 to MT90502

Streams are assigned to this mode in groups of 4 only.

n = 0, 3, 7, 11, ....

**Figure 9 - PCM/ADPCM Data Format B**

**Format B for RX xxPCM Samples**
**(with Auto PCM-ADPCM Switching)**

**Format of PCM Samples**
**(for both padding and voice)**
**(with Decompressor reset indicated)**

| PCM-R | | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | PCM Sample | | | | | | | |

**Format of PCM Samples**
**(for both padding and voice)**
**(no Decompressor reset indicated)**

| PCM-R | | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | PCM Sample | | | | | | | |

**Format of ADPCM Samples in low bits**
**(with Decompressor reset indicated)**

| PCM-R | | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | ADPCM 40 kbps | | | | |
| 0 | 1 | 1 | 0 | 0 | 1 | 32 kbps | | | |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 24 kbps | | |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 16 kbps | |

**Format of ADPCM Samples in low bits**
**(no Decompressor reset indicated)**

| PCM-R | | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | ADPCM 40 kbps | | | | |
| 0 | 0 | 0 | 0 | 0 | 1 | 32 kbps | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 24 kbps | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 16 kbps | |

**Format of ADPCM Samples in high bits**
**(with Decompressor reset indicated)**

| PCM-R | | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | ADPCM 40 kbps | | | | | 1 | 0 | 1 |
| 0 | 1 | 32 kbps | | | | 1 | 0 | 0 | 1 |
| 0 | 1 | 24 kbps | | | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 16 kbps | | 1 | 0 | 0 | 0 | 0 | 1 |

**Format of ADPCM Samples in high bits**
**(no Decompressor reset indicated)**

| PCM-R | | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | ADPCM 40 kbps | | | | | 1 | 0 | 0 |
| 0 | 0 | 32 kbps | | | | 1 | 0 | 0 | 0 |
| 0 | 0 | 24 kbps | | | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 16 kbps | | 1 | 0 | 0 | 0 | 0 | 0 |

**Figure 10 - PCM/ADPCM RX Data Format B**

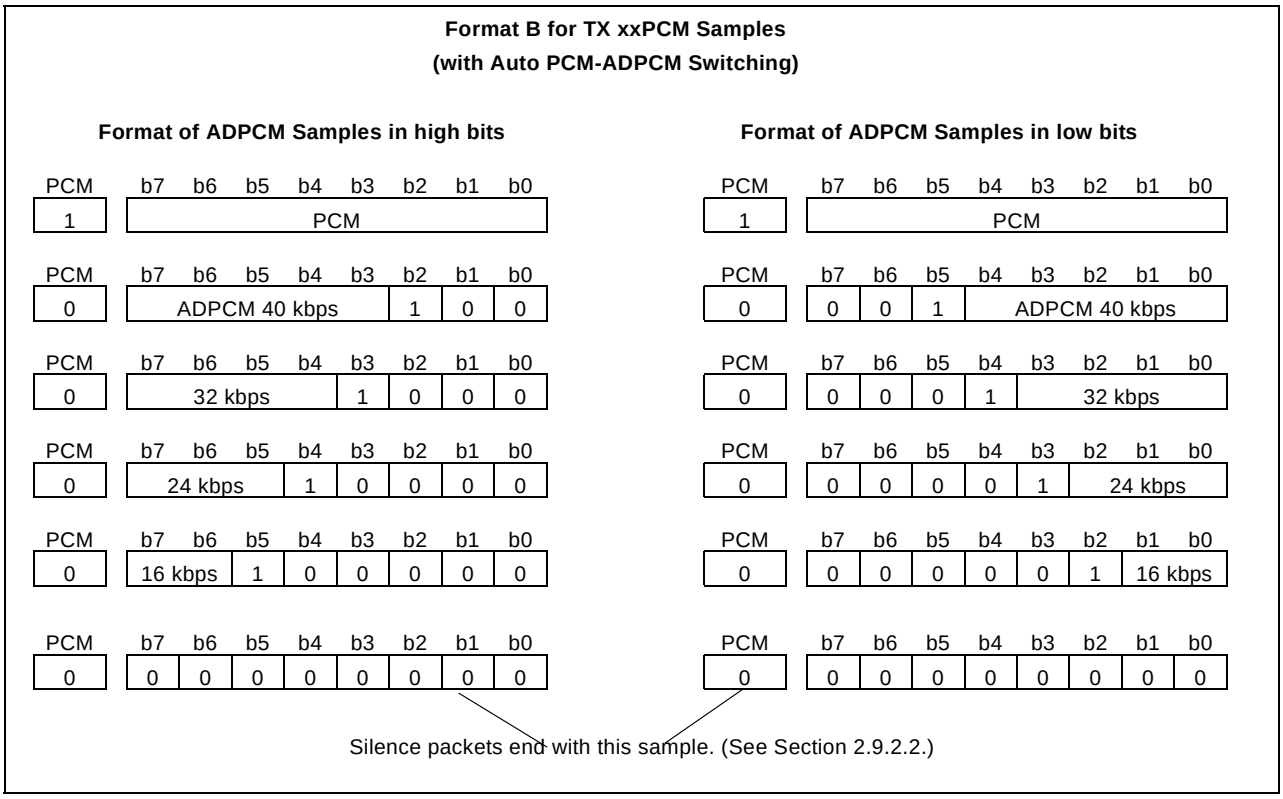



**Figure 11 - PCM/ADPCM TX Data Format B**

### 2.2.2.3 Phase Alignment

In the MT90502 CPS-Packet assembly engine, any given frame can begin a CPS-Packet. Once this frame is established, packets will always be formed every 8 x #EDU frames. In order to control the trade-off between delay and bandwidth efficiency, the MT90502 supports phasing of CPS-Packet construction. For any given EDU value the MT90502 selects a frame as phase 0 and sub-phase 0. Phase specifies the EDU relative to phase 0 and sub-phase specifies a sample offset within the EDU. For example if a 5 EDU connection opened with a phase of 1 and sub-phase of 2 then it will begin assembly 10 frames after a connection of phase 0, sub-phase 0.

Internal phase and sub-phase counters exist to co-ordinate the accumulation of sufficient data for a CPS-Packet with the construction of CPS-Packets. There are multiple independent phase counters, one for each #EDU. Channels with the same setting of #EDU share the same phase counter. Phase counters count from 0 to 1, 2, 3, 4, 7, 9 and 10. One global sub-phase counter exists which counts from 0 to 7 and increments on every frame. Each time the sub-phase counter reaches 0, the phase counters increment. A CPS-Packet is created for a channel when the channel's phase counter equals the channel's phase number, and the sub-phase counter equals the channel's sub-phase number. Both phase and sub-phase numbers are user programmed in the PCM/ADPCM CPS-Packet Assembly Structure, shown in Figure 12.

The MT90502 has the capability of aligning the transmission of PCM and ADPCM CPS-Packets containing the same number of EDUs and destined to the same VC in order to obtain less delay and higher bandwidth efficiency. To accomplish this, the phase and sub-phase fields of the PCM/ADPCM CPS-Packet assembly structure must match the phase and sub-phase field of all other channels that are to be phase aligned on the same VC. Programming the phase and sub-phase fields is done by the CPU upon configuration of the CPS-Packet assembly structure.

By default, all phase and sub-phase counters are free running. Each counter picks up an arbitrary frame as phase 0 or sub-phase 0, then starts its counting. The phasing information, e.g. which frame is phase 0 and sub-phase 0, is internal to the device in this case.

For certain applications some external devices must be aware of what phase and sub-phase the current frame represents, so that they can know where the CPS-Packet boundary is. An example is ADPCM compressor which should only change its compression rate at CPS-Packet boundaries. To indicate a CPS-Packet boundary a phasing TSST structure is introduced to allow an external device to send phasing information to the MT90502. Once a TSST is configured as phasing channel in TX CAM, the data on that TSST will overwrite internal counter(s). When the MT90502 sees a 00h on that TSST, it takes that frame as phase 0 and sub-phase 0. When there is a 01h, it interprets it as phase 0 sub-phase 1, and so on. The external device driving a phasing channel must ensure that a proper counting pattern appears on this channel every frame.

Up to four phasing channels can be created, each with a different XPI value. When loading data to internal counters, extra MSBs will be truncated. For example using a XPI value of ‘10’, a counting pattern of 0 to 63 also serves as two cycles of 0 to 31, during which two 4-EDU CPS-Packets will be assembled.

| XPI | Counting Pattern | Replacing Internal Counters |
|---|---|---|
| 00 | 0 to 23 | #EDU = 3 |
| 01 | 0 to 39 | #EDU = 5, and 40/80 frame PCM/ADPCM |
| 10 | 0 to 63 | #EDU = 1, 2, 4, 8 |
| 11 | 0 to 87 | #EDU = 44/88 frame PCM/ADPCM |

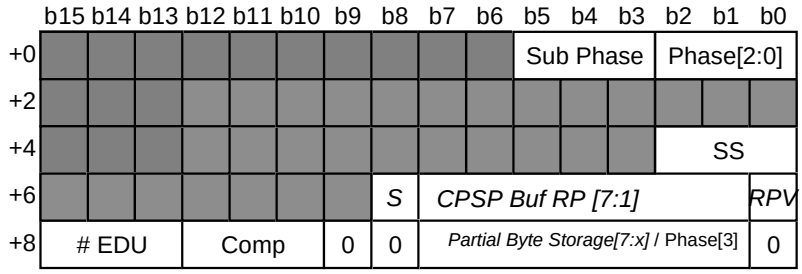
**Table 18 - XPI Selection Table**

### 2.2.2.4 PCM/ADPCM CPS-Packet Assembly Structure

PCM/ADPCM CPS-Packet Assembly Structures are located in internal TX TDM Control Memory. Each PCM or ADPCM channel will find an entry to its CPS-Packet Assembly Structure by the xxPCM Channel Number assigned in the TX CAM.

**TX TDM Control Memory**

PCM/ADPCM CPS-Packet Assembly Structure




Fields in *Italic* are used by hardware only.

Fields in Plain are written to by the CPU/Software.

Index of structure given by PCM Channel number in TX CAM.

Reserved

Structures are located in internal TX TDM Control Memory (8000h to BFF0h).

Each structure is 16-bytes in size.

Note: The write pointer to the xxPCM Circular Buffers is global. Different write pointers exist for each compression type and phase.

**#EDU**: The number of EDUs per PCM CPS-Packet assembled. “000” = 1 EDU, “001” = 2 EDUs; “010” = 3 EDUs; “011” = 4 EDUs; “100” = 5 EDUs; “101” = 44/88 frame PCM/ADPCM; “110” = 40/80 frame PCM/ADPCM; “111” = 8 EDUs.

**Compression**: “000” = PCM; “001” = ADPCM 40 kbps; “010” = ADPCM 32 kbps; “011” = ADPCM 24 kbps; “100” = ADPCM 16 kbps; “101” = ADPCM auto-detect; “110” = PCM & ADPCM auto-detect (TDM Format B must be chosen); “111” = reserved.

**Partial Byte Storage[7:x] / Phase[3]**: Used by hardware to store ADPCM samples that have not formed complete bytes yet. When #EDU indicates 44/88-frame or 40/80 frame mode, the partial byte requires only 4 bits (ADPCM-32) and is stored in bits 7:4. The LSB contains Phase[3] (programed by the CPU/Software).

**CPSP Buf** : TX SAR’s pointer of up to where it has read in the channel’s circular buffer. Used to detect overflows.

**RPV** : Indicates that the CPSP Buf RP is valid, and thus that Circular Buffer Overflows can be detected. Written to ‘1’ any time the CPSP Buf RP is written.

**S**: Silent Bit. Indicates if the current CPS-Packet is silent so far or not.

**SS**: Silence Suppression. “000” = no silence suppression; “001” = associated odd stream bit represents silence; “010” = A Match; “011” = B Match; “100” = silence indicated as in Fig 10; others = reserved

**Sub Phase**: Phase alignment of the channel within the 8 sub-phases possible in an EDU.

**Phase[3:0]**: Phase alignment of the channel among the different EDUs it may contain. This number ranges between 0 and #EDU field. When #EDU is 44/88-frame packets are assembled every 44 or 88 frames. Phase and sub-phase specify where assembly should start for 88 frame packets and when assembling 44 frame packets one will begin on the specified phase/sub-phase and another will begin 44 frames later by default. Therefore, there are eleven (0-10) phases in 44/88 frame mode. Similarly in 40/80 frame mode there are ten (0-9) phases. Bit 3 of the phase is encoded in Partial Byte Storage[1].

**Figure 12 - PCM/ADPCM CPS-Packet Assembly Structure**

### 2.2.3 Treatment of HDLC Data

Other than PCM and ADPCM data, MT90502 can also send/receive generic AAL2 CPS-Packets directly to/from the H.100/H.110 bus. Those CPS-Packets must be encapsulated in HDLC format before they can be put onto H.100/H.110 bus. The HDLC encapsulation of the data allows an external SSCS engine (e.g. G.723, G.728 or G.729) to pass over CPS-Packets or CPS-Packet payloads to the MT90502 through TDM bus.

HDLC data is accumulated until a flag is received signifying the end of an HDLC packet (see Section 2.10 on page 94). Then, either bit-wise or byte-wise zero-extraction (configurable for the entire chip in register 500h) is performed on the HDLC data. At this stage, the HDLC packets can be re-formatted as AAL2 CPS-Packets for treatment by the TX SAR.

#### 2.2.3.1 HDLC Streams

Any number from 1 to 128 of consecutive time slots on a single H.100/H.110 stream can be grouped as a single HDLC stream to carry HDLC packets. For each TSST, an entry in TX CAM lists the HDLC stream number associated with that TSST; these stream numbers are the same for all TSSTs carrying data for a single HDLC stream. Within a single HDLC stream, one or more HDLC channels can be transported. An HDLC channel carries CPS-Packets destined to the same AAL2 channel. Up to 1023 channels, whether they are xxPCM channels or HDLC channels, are supported by MT90502.

Each HDLC stream must be assigned N consecutive assembly structures in TX TDM Control Memory, as seen in Figure 13, “HDLC CPS-Packet Assembly Structure,” on page 33. N is always the same as the number of HDLC channels carried by that HDLC stream with only one exception, that is, when the number of HDLC channels is one, and the TSSTs occupied by the HDLC stream is more than one, N must be 2.

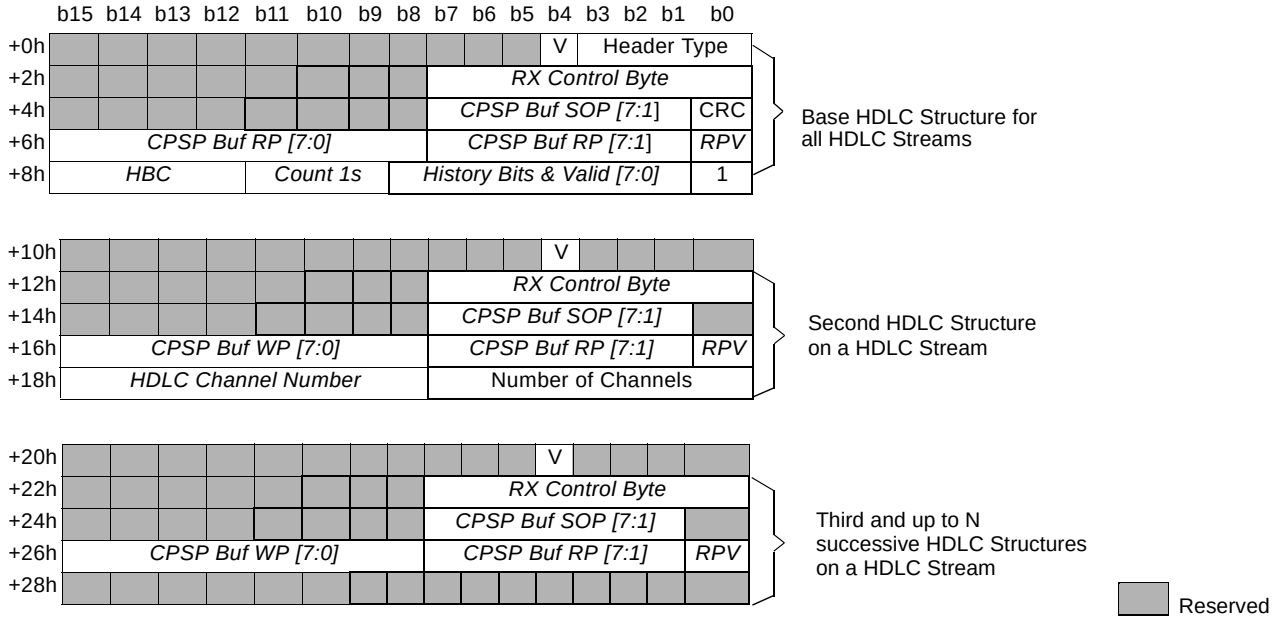
#### 2.2.3.2 Address Bytes

HDLC packets may contain zero, one, or two address bytes that immediately follow the start flag. For HDLC packets containing one or two address bytes, the last address byte can be used to indicate a channel number. If the last address byte is used to indicate a channel number, each of the 256 possible channel numbers can be directed to a different CPS-Packet final assembly structure. Each CPS-Packet, based on its HDLC channel number, can be directed to a different VC/CID. If the value of the last address byte exceeds the allowed number of HDLC channels for that HDLC stream, the channel number 0 will be used for that CPS-Packet. All address bytes not used to indicate a channel number are discarded by the TX TDM module and do not appear in an AAL2 CPS-Packet. The HDLC stream number added to the HDLC address byte will yield the HDLC channel number.

Whether or not the address byte(s) will be used to indicate a channel number is determined independently for each HDLC stream using the “Header Type” field of the HDLC CPS-Packet assembly structure (see Figure 13 on page 33).

**TX TDM Control Memory (AAL2 header not present in HDLC data)**

**HDLC CPS-Packet Assembly Structure**




Fields in *Italic* are used by hardware only. Fields in Plain are written to by the CPU/Software.

Index of the base structure given by HDLC Stream Number in TX CAM.

Structures located in internal TX TDM Control Memory (8000h to BFF0h).

Each structure is 16-bytes in size.

**V:** Valid. When 0, this HDLC channel will be ignored. When 1, it will be processed. This valid bit has no effect on the operation of the HDLC stream.

**Header Type:** Header type. 0000 = Plain; 0001 = One byte address; 0010 = Two byte address; 0011 = One byte address + Control; 0100 = Two byte address + Control; 0101 = One byte address (SCS); 0110 = Two byte address (SCS); 0111 = One byte address + Control (SCS); 1000 = Two byte address + Control (SCS).

SCS: Sub-Channel Scheme where address byte is used to differentiate HDLC channels. Address bytes are ignored in non-SCS modes.

**CRC:** 0 = CRC not present; 1 - 16-bit CRC present.

**CPSP Buf WP:** CPS-Packet Buffer Write Pointer. Points to the next byte that will be written in the CPS-Packet Circular Buffer.

**CPSP Buf RP[7:1]** Pointer to the next word that will be read by the TXSAR's Cell Assembly Process. If a byte is written at this word, an error will be flagged.

**RPV:** Indicates that the CPSP Buf RP is valid, and that Circular Buffer Overflows can be detected. Written to 1 any time the CPSP Buf RP is written.

**CPSP Buf SOP[7:1]** Pointer to the start of the current CPS-Packet (being assembled now).

**HBC:** HDLC Byte Count. Counts the first few bytes of each CPS-Packet. This counter will not increment beyond '1000'. It will be reset upon the reception of a flag.

**RX Control Byte:** Control byte received in the last HDLC packet.

**History Bits & Valid:** Eight history bits used to do zero extraction and to indicate the validity of the received data. 00000001 = no leftover bits, 1xxxxxxx=7 leftover bits. Should be initialized to 00000001.

**HDLC Channel Number:** HDLC Channel Number of the Current HDLC Packet being received.

**Number of Channels:** Number of channels in this HDLC Stream. Valid range is 0 to 255. 0 means 256.

**Figure 13 - HDLC CPS-Packet Assembly Structure**

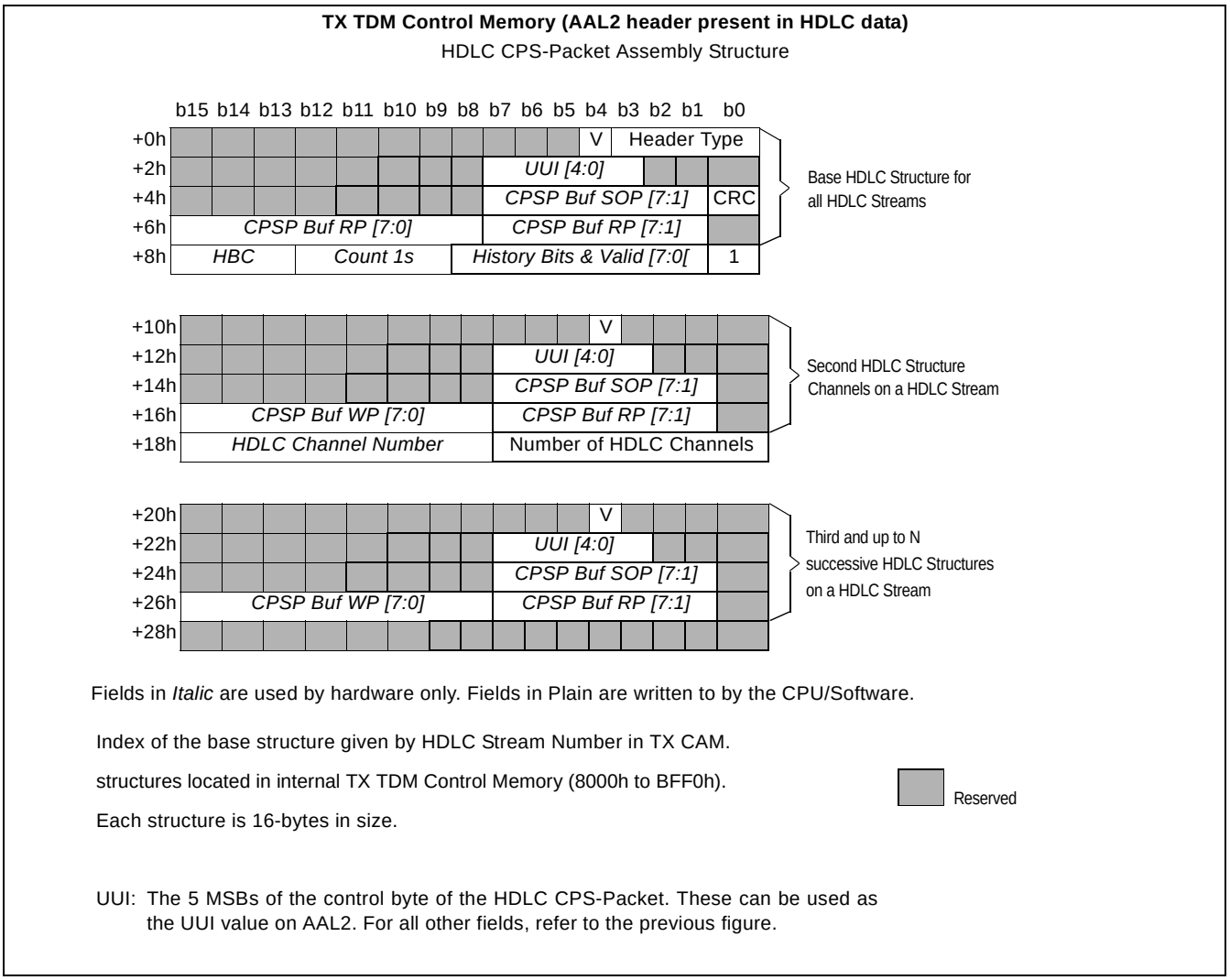



**Figure 14 - HDLC CPS-Packet Assembly Structure**

### 2.2.3.3 Control Bytes and Length

In addition to a zero, one, or two byte address, an HDLC packet may also contain a control byte and a two-byte CRC. The 5 MSBs of the HDLC control byte can be used as the UUI in the AAL2 CPS-Packet formed from that HDLC packet.

The TX TDM calculates the length of each HDLC packet and stores it in the Length field of the each TX TDM Write Cache entry.

### 2.2.3.4 “Raw” AAL2 CPS-Packets

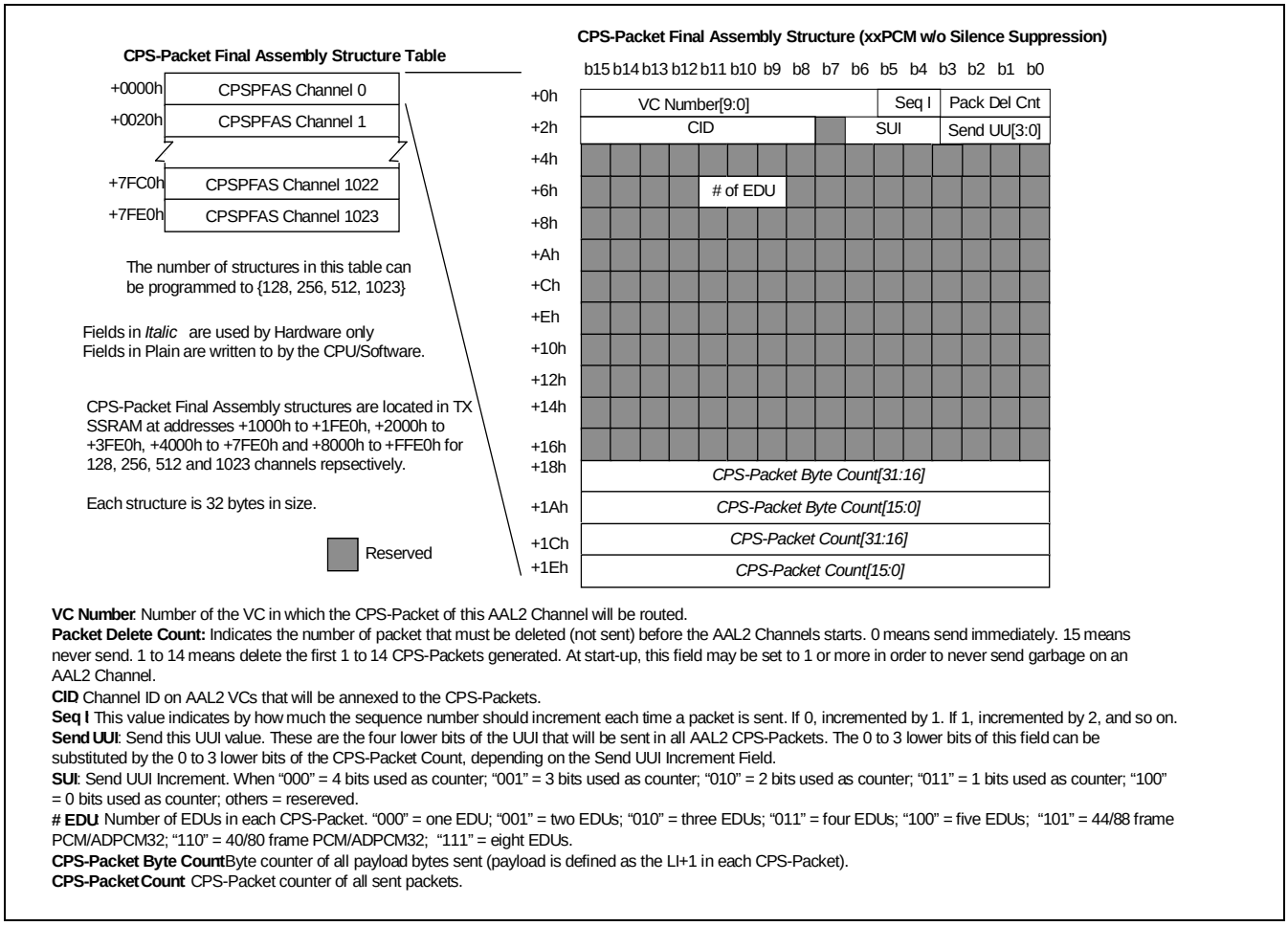
The data field of an HDLC packet can contain an AAL2 CPS-Packet with or without a header. This is configured for all HDLC streams in register 500h. When the packaging_type field of register 500h is set to ‘1’, all HDLC packets will carry complete CPS-Packets with both header and payload. They will be treated as “raw” AAL2 CPS-Packets.

For “raw” AAL2 CPS-Packets, the CID is ignored and replaced by the CID in the CPS-Packet final assembly structure. The LI is ignored and re-calculated by the TX TDM module. The UUI is passed on as the UUI for the AAL2 CPS-Packet. The HEC is ignored and regenerated based on the new CID/UUI/LI.

### 2.2.4 CPS-Packet Final Assembly

Once PCM/ADPCM and HDLC data is segmented into CPS-Packets, the CPS-Packet final assembly structure is referenced and the packaging is completed through the addition of the CID, UUI, LI and HEC. Finally, the CPS-Packets are directed to the correct location in the CPS-Packet Buffer (according to the channel number for that CPS-Packet) from which the TX SAR will read them.

The UUI can be sourced from a “raw” AAL2 CPS-Packet, the 5 MSBs of the control byte of an HDLC packet, or, the LSBs can be a one-bit to four-bit free-running counter. When the UUI is used as a counter, the MSB must be 0.

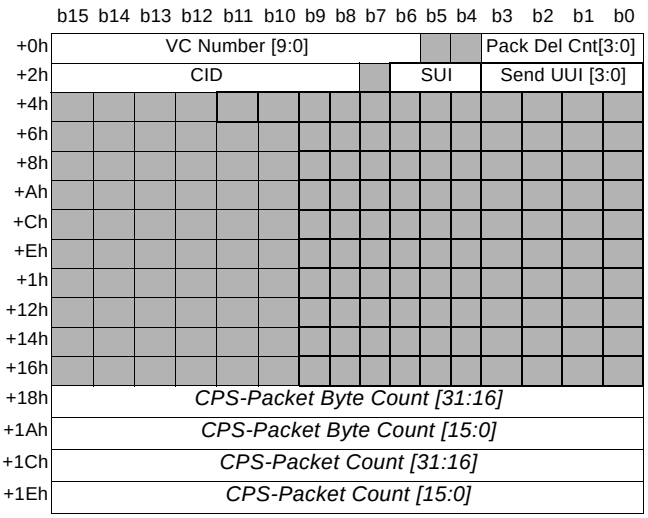



**VC Number**: Number of the VC in which the CPS-Packet of this AAL2 Channel will be routed.
**Packet Delete Count:** Indicates the number of packet that must be deleted (not sent) before the AAL2 Channels starts. 0 means send immediately. 15 means never send. 1 to 14 means delete the first 1 to 14 CPS-Packets generated. At start-up, this field may be set to 1 or more in order to never send garbage on an AAL2 Channel.
**CID**: Channel ID on AAL2 VCs that will be annexed to the CPS-Packets.
**Seq I**: This value indicates by how much the sequence number should increment each time a packet is sent. If 0, incremented by 1. If 1, incremented by 2, and so on.
**Send UUI**: Send this UUI value. These are the four lower bits of the UUI that will be sent in all AAL2 CPS-Packets. The 0 to 3 lower bits of this field can be substituted by the 0 to 3 lower bits of the CPS-Packet Count, depending on the Send UUI Increment Field.
**SUI**: Send UUI Increment. When “000” = 4 bits used as counter; “001” = 3 bits used as counter; “010” = 2 bits used as counter; “011” = 1 bits used as counter; “100” = 0 bits used as counter; others = resereved.
**# EDU**: Number of EDUs in each CPS-Packet. “000” = one EDU; “001” = two EDUs; “010” = three EDUs; “011” = four EDUs; “100” = five EDUs; “101” = 44/88 frame PCM/ADPCM32; “110” = 40/80 frame PCM/ADPCM32; “111” = eight EDUs.
**CPS-Packet Byte Count**Byte counter of all payload bytes sent (payload is defined as the LI+1 in each CPS-Packet).
**CPS-Packet Count**: CPS-Packet counter of all sent packets.

**Figure 15 - CPS-Packet Final Assembly Structure (PCM/ADPCM)**

The CPS-Packet final assembly structure contains two monitoring fields: a 32-bit byte count and a 32-bit CPS-Packet count. The CPS-Packet count indicates the number of complete CPS-Packets that have been generated using this structure, and the byte count indicates the number of data bytes contained in all those CPS-Packets, excluding the AAL2 headers.

**CPS-Packet Final Assembly Structure (HDLC Channel)**




Fields in *Italic* are used by hardware only. Fields in Plain are written to by the CPU/Software.

CPS-Packet Final Assembly structures are fixed and are located in TX SSRAM memory at addresses +1000h to +1FE0h, +2000h to +3FE0h, +4000h to +7FE0h and +8000h to +FFE0h for 128, 256, 512 and 1023 channels respectively.

Each structure is 32-bytes in size.

**VC Number:** Number of the VC in which the CPS-Packets of this AAL2 Channel will be routed.

**Packet Delete Count:** Indicates the number of packets that must be deleted (not sent) before the AAL2 Channel starts. 0 means send immediately. 15 means never send. 1 to 14 mean delete the first 1 to 14 first CPS-Packets generated. At start-up, this field may be set to 1 or more in order to never send garbage on an AAL2 Channel.

**CID:** Channel ID on AAL2 VCs that will be annexed to the CPS-Packets.

**Send UUI:** Send this UUI value. These are the four lower bits of the UUI that will be sent in all AAL2 CPS-Packets. The 0 to 4 lower bits of this field can be substituted by the 0 to 4 lower bits of the CPS-Packet Count, depending on the Send UUI Increment Field.

**SUI:** Send UUI Increment. When “000” = 4 bits used as counter, “001” = 3 bits used as counter, “010” = 2 bits used as counter, “011” = 1 bit used as counter, “100” = 0 bits used as counter, “101” = send the UUI value contained in the HDLC CPS-Packet control byte. Others = reserved. Once in “raw” CPS-Packet mode, Send UUI and SUI are ignored.

**CPS-Packet Byte Count:** Byte counter of all payload bytes sent (payload is defined as the LI+1 in each CPS-Packet).

**CPS-Packet Count:** CPS-Packet counter of all sent packets.

**Figure 16 - CPS-Packet Final Assembly Structure (HDLC Channel)**

### 2.2.4.1 CPU CPS-Packets

The TX TDM module allows the generation of CPU sourced CPS-Packets. When the CPU requires to transmit an AAL2 CPS-Packet onto a certain VC, it programs the VC number in register 520h, and writes the CPS-Packet's descriptor structure in registers 522h to 526h. The format of the CPS-Packet descriptor structure is given in Figure 17. Word 0 must be written into register 522h, Bytes +2 and +5 to register 524h and Bytes +4 and +5 to register 526h. Finally, it writes a '1' to the cps_packet_request bit in register 520h, which begins the request. When the bit clears, the process has concluded and the CPS-Packet has been added to the TX VC's queue.

The following process is used to send a CPU CPS-Packet:

1) The payload of the CPS-Packet must be written into some unused portion of the external SSRAM (bank A). The format for writing the payload to memory is shown in the following example. In the example, the CPS-Packet payload is 25-bytes.

| address | bits [15:8] | bits[7:0] |
|---|---|---|
| cpsp_base + 0 (word 0): | payload 0 | payload 1 |
| cpsp_base + 2 (word 1): | payload 2 | payload 3 |
| ... | ... | ... |
| cpsp_base + 24 (word 12): | payload 24 | xxxxxxxx |

**Table 19 - Format for writing CPS-Packet payload to memory**

2) A CPS-Packet descriptor is written to registers 522h, 524h, and 526h. The value to which each field must be set is indicated in Figure 17, “CPS-Packet Descriptor Queue,” on page 38.

CPS-Packet base address: Base address of CPS-Packet’s payload bytes. The address is relative to the base address of bank A. Thus, if payload bytes 0 and 1 of the CPS-Packet are located at address 400126h, then the address inserted is 00126h.

WB: Set to ‘1’. Write back must be disabled for CPU sourced CPS-Packets.

CID: The value of the CID of the sent CPS-Packet. Any 8-bit value is valid, with the exception of 00h, which is illegal (AAL2 specification).

CPU: Set to ‘1’. Indicates that the CPS-Packet is sourced by the CPU.

Size: As mentioned earlier, the payload of the TX CPU CPS-Packets can be inserted in any unoccupied portion of SSRAM bank A. A portion of bank A may be reserved for a virtual buffer. This buffer may be used to store the payload bytes of all CPU sourced CPS-Packets. In the case where the payload of a CPS-Packet is inserted at the end of the buffer and wraps to the beginning, the Size field indicates where to wrap to. Take for example a CPS-Packet of 10 payload bytes, and the Size field set to “0000” (256 bytes). Thus, the CPS-Packet must reside in a 256 byte boundary of the external memory. Now suppose that the CPS-Packet is written to address 1FEh. This implies that the payload bytes must be written as follows:

| address | bits [15:8] | bits[7:0] |
|---|---|---|
| 100h (word 1): | payload 2 | payload 3 |
| 102h (word 2): | payload 4 | payload 5 |
| 104h (word 3): | payload 6 | payload 7 |
| 106h (word 4): | payload 8 | payload 9 |
| ... | ... | ... |
| 1FEh (word 0): | payload 0 | payload 1 |

**Table 20 - Example of written payload bytes**

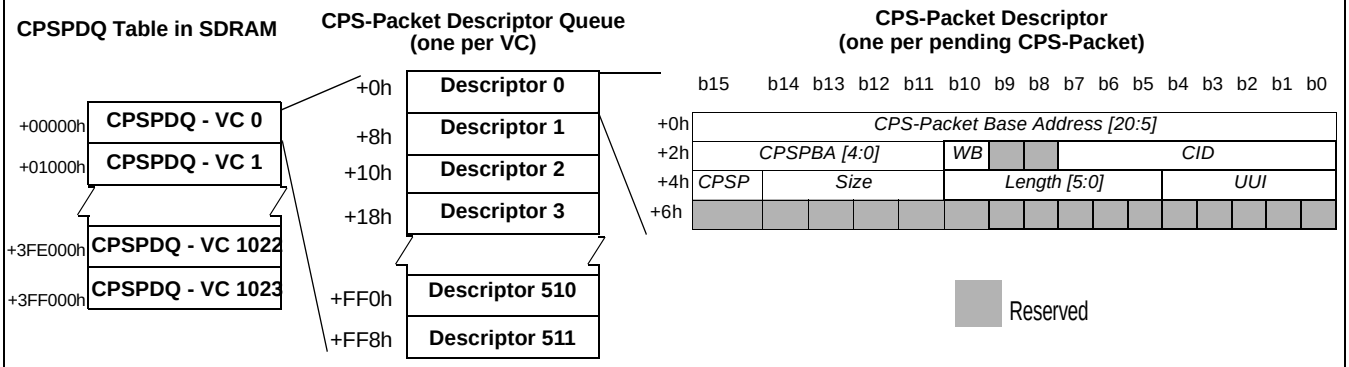
Length: Set to the length of the CPS-Packet, minus 1 (i.e. LI field).

UUI: The value of the UUI field of the sent CPS-Packet. Any 5-bit value is valid.

Note: CPU CPS-Packets sent out on voice CIDs must use UUIs 16-31. They can use all UUIs on management CIDs (1-7).

3) Once the descriptor is written to the registers, the TX TDM module of the MT90502 is made aware of the CPS-Packet through register 520h. The VC number of an open AAL2 VC is written in bits [9:0], and the request bit, bit [10] of the register, is set to ‘1’. Finally, the register is written to. The CPU CPS-Packet has been written when the request bit of the register is cleared by hardware.

### 2.2.4.2 CPS-Packet Descriptor Queue




The number of CPSPDQ in this table can be programmed to {128, 256, 512, 1023}.

CPS-Packet Descriptor structures are fixed and are located at the beginning of SDRAM memory (2000000h).

Each structure is 8-bytes in size.

Note: This structure is used by the hardware only.

**CPSPBA:** CPS-Packet Base Address [20:0]: Word pointer pointing to the beginning of a CPS-Packet. Note that the CPS-Packet as such wraps at the boundary of its circular buffer.

**Size:** Indicates the "wrap" boundary of the CPS-Packet. "0000" = 256 bytes; "0001" = 256 x 2^1; "0002" = 256 x 2^2; ... "1101" = 2M bytes.

**CID:** Channel ID that will be passed on in the AAL2 CPS-Packet.

**Length**: Length of the data part of the packet. Defined in the same way as the LI.

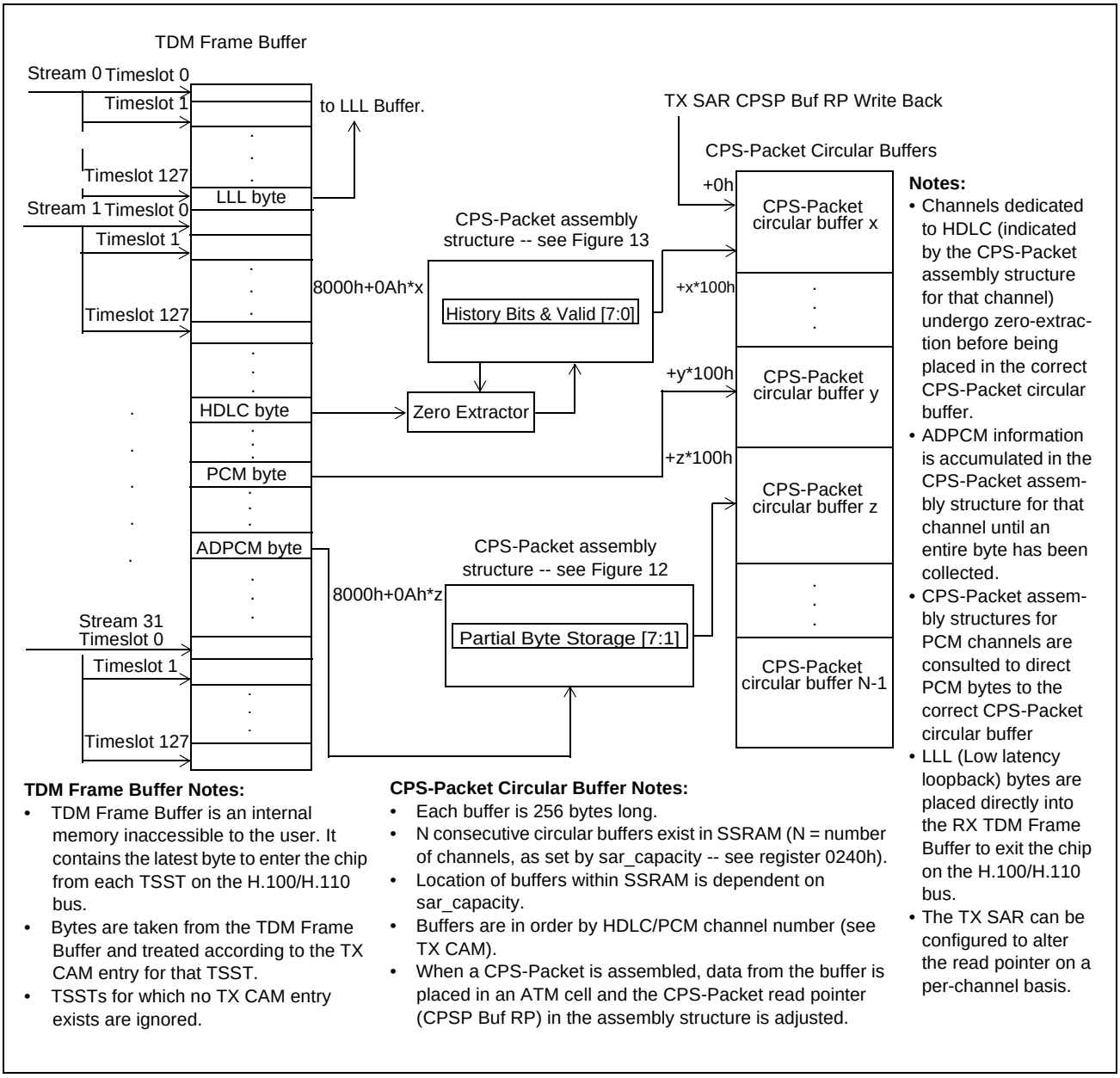
**UUI:** User to User indicator that will be passed on in the AAL2 CPS-Packet.

**CPU:** CPU Sourced CPS-Packet. "0" = CPS-Packet from TDM bus; "1" = CPS-Packet from CPU.

**WB:** When "1", the TX RX SAR will NOT write back the CPSP Read Pointer in the TXTDM PCM Control Structure.

**Figure 17 - CPS-Packet Descriptor Queue**

### 2.2.4.3 TDM Frame Buffer




**TDM Frame Buffer Notes:**

- TDM Frame Buffer is an internal memory inaccessible to the user. It contains the latest byte to enter the chip from each TSST on the H.100/H.110 bus.
- Bytes are taken from the TDM Frame Buffer and treated according to the TX CAM entry for that TSST.
- TSSTs for which no TX CAM entry exists are ignored.

**CPS-Packet Circular Buffer Notes:**

- Each buffer is 256 bytes long.
- N consecutive circular buffers exist in SSRAM (N = number of channels, as set by sar_capacity -- see register 0240h).
- Location of buffers within SSRAM is dependent on sar_capacity.
- Buffers are in order by HDLC/PCM channel number (see TX CAM).
- When a CPS-Packet is assembled, data from the buffer is placed in an ATM cell and the CPS-Packet read pointer (CPSP Buf RP) in the assembly structure is adjusted.

**Notes:**

- Channels dedicated to HDLC (indicated by the CPS-Packet assembly structure for that channel) undergo zero-extraction before being placed in the correct CPS-Packet circular buffer.
- ADPCM information is accumulated in the CPS-Packet assembly structure for that channel until an entire byte has been collected.
- CPS-Packet assembly structures for PCM channels are consulted to direct PCM bytes to the correct CPS-Packet circular buffer
- LLL (Low latency loopback) bytes are placed directly into the RX TDM Frame Buffer to exit the chip on the H.100/H.110 bus.
- The TX SAR can be configured to alter the read pointer on a per-channel basis.

**Figure 18 - TDM Frame Buffer**

## 2.3 TX SAR

### 2.3.1 Overview

The purpose of the TX SAR Module is to assemble AAL2 CPS-Packets into AAL2 ATM cells to be transmitted to the UTOPIA Module. The TX SAR Module has no external interfaces, and does not control any of the MT90502's pins. However, it is the connecting block between the Tx TDM module and the UTOPIA Module in the data transmission direction.

### 2.3.2 AAL2 Cell Assembly Process

The AAL2 cell assembly process is directed by the TX AAL2 VC Structure (see Figure 19 on page 40). This structure is referenced and references all the structures necessary to compile an AAL2 ATM cell. It is employed by the TX SAR module to raise an event for an AAL2 ATM cell. It constructs the ATM cell payload by referencing the CPS-Packets via the CPS-Packet Descriptor Queue (SDRAM), which contains the CPS-Packet addresses in the CPS-Packet Circular Buffer (TX SSRAM). The structure also contains the ATM header data, time-out data, next CPU transmit CPS-Packet pointer, UTOPIA port destination and a free running counter of transmitted ATM cells.

It is the responsibility of the software to program the following in the TX AAL2 VC Structure (Figure 19).

- time-out period
- ATM header associated with the VC
- the TXD (TX Destination) field

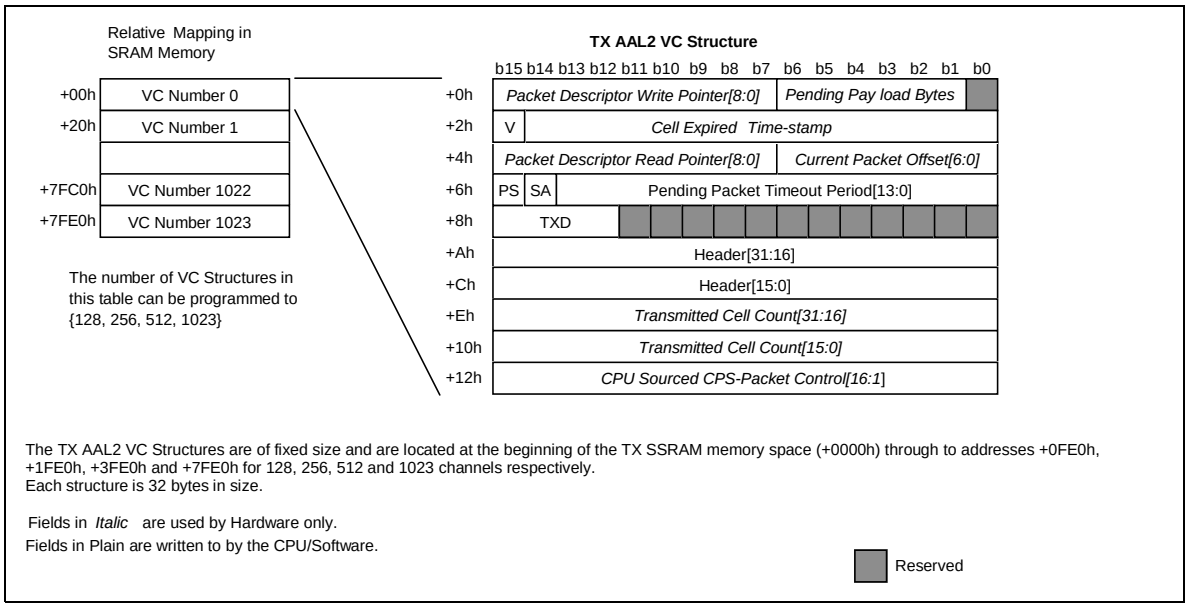



**Figure 19 - TX AAL2 VC Structure**

| Field | Byte Address Offset/Bits Used | Description of Field |
|---|---|---|
| Packet Descriptor Write Pointer | 0/b[15:7] | Written by the TX TDM module to denote the address of next CPS-Packet Descriptor in the CPSPDQ. The CPSPDQ is deemed full when the Packet Descriptor Write Pointer is one less than the Packet Descriptor Read Pointer. |
| Pending Payload Bytes | 0/b[6:1] | Indicates the number of pending payload bytes (CPS-Packet payload and header bytes) available to the associated VC. The range is 0 to 46. This field is updated by the TX TDM process with the remaining ATM cell pending payload bytes when a Cell Assembly Event is raised. Reset is performed by the TXSAR's scanning process. |
| V: Valid Bit | +2/b[15] | Associated to the Cell Expired Time-Stamp. If this bit is '1', then the time-stamp must be taken into account. Otherwise, it is ignored and no cells can be assembled. |
| Cell Expired Time Stamp | +2/b[14:0] | This is an absolute time stamp at which a pending cell must be transmitted. This field is set by the TDM process any time a CPS-Packet is written that starts at or overlaps a cell boundary. It will be written to the current time plus the Pending Packet Time-out Period field in this structure. This field will be cleared by the TDM process any time no CPS-Packets are pending. It will also be cleared by the TXSAR's scanning process if it is detected to have timed-out, and a cell Transmission Event is scheduled by this same process. The time unit for the time-out is 125 µs. |
| Packet Descriptor Read Pointer | +4/b[15:7] | Read pointer to the current CPS-Packet (Descriptor) in the CPSPDQ. Updated by the TXSAR. |
| Current Packet Offset | +4/b[6:0] | This field indicates how many bytes are remaining in a CPS-Packet when the CPS-Packet is straddling two cells. |
| PS: Prevent Scanning | +6/b[15] | When this bit is '1', the pending packet time-out period will be ignored, and the V bit will never be written back to '1'. |
| SA: Send Always | +6/b[14] | When '1', a maximum of one CPS-Packet will be contained in each cell. The remainder of the last cell to contain a CPS-Packet will always be zero padded to make sure that no other CPS-Packets join it. |
| Pending Packet Time-out Period | +6/b[13:0] | This is the maximum period set by the software that a Pending CPS-Packet exist before transmission. The value is in multiples of 125µs. 0 means between 0 and 124µs. |
| TXD: Transmission Destination | +8/b[15:12] | 000 do not send<br>0xx1 UTOPIA TX Port A<br>0x1x UTOPIA TX Port B<br>01xx UTOPIA TX Port C<br>1000 UTOPIA RX Port A<br>1001 UTOPIA RX Port B<br>1010 UTOPIA RX Port C<br>others reserved |
| Header[31:0] | +A/b[15:0]<br>+C/b[15:0] | ATM Cell Header in the following order (starting from bit 31): GFC, VPI, VCI, PT, CLP |

**Table 21 - AAL2 VC Structure Fields**

| Field | Byte Address Offset/Bits Used | Description of Field |
|---|---|---|
| Transmitted Cell Count | +E/b[15:0]<br>+10/b[15:0] | Free running transmitted cell counter. |
| CPU Sourced CPS-Packet Control | +12/b[15:0] | This field points to the first word of the next CPU sourced CPS-Packet. |

**Table 21 - AAL2 VC Structure Fields (continued)**

#### 2.3.2.1 AAL2 Cell Assembly Procedure

- The CPS-Packet Assembly Structure raises a CPS-Packet event upon the alignment of the sub-phase and phase (See Figure 12 - PCM/ADPCM CPS-Packet Assembly Structure).
- The CPS-Packet Assembly Structure is mapped to the CPS-Packet Final Assembly Structure by the absolute address of both structures.
- The CPS-Packet Final Assembly Structure reads the ‘CPS-Packet Descriptor Write Pointer’ in the AAL2 VC Structure to determine the next address to write the current CPS-Packet.
- The CPS-Packet Assembly Structure retrieves the payload data from the TX TDM Frame Buffer and writes it to the CPS-Packet Circular Buffer in TX SSRAM.
- In preparation for the next CPS-Packet, the CPS-Packet Final Assembly Structure then updates (i.e. writes) the ‘Packet Descriptor Write Pointer’ in the ‘AAL2 VC Structure’.
- The ‘Pending Payload Byte’ field is updated by the TX TDM module when a CPS-Packet event is raised.
- When more than one CPS-Packet event is raised within the same CT_FRAME, each CPS-Packet event generates its corresponding CPS-Packets Final Assembly Structure in the TX TDM process. The TX SAR sequentially steps through each CPS-Packet Final Assembly Structure and for each process consults the associated AAL2 VC Structure to determine if enough payload has accumulated to generate a VC Send Event (see Figure 20, “Cell Assembly Event Queue,” on page 43). If enough payload is present then the TX SAR raises a VC Send Event and updates the Pending Payload Byte field with the remaining number of pending payload bytes. However, a VC Send Event may also be generated via the time-out period specified in the AAL2 VC Structure. All VCs have a time-out period that specifies the maximum time that any CPS-Packet destined to that VC can wait before a cell containing them is assembled. The time-out period is defined in CT Frames (i.e. multiples of 125µs) in the AAL2 VC Structure.
- The ATM header information is contained within the AAL2 VC Structure.
- The Packet Descriptor Read Pointer in the AAL2 VC Structure points indirectly to the complete CPS-Packet via the CPS-Packet Base Address in the Descriptor of the CPS-Packet Descriptor Queue.
- The CPS-Packet Descriptor Queue contains Descriptors for each complete CPS-Packet. The Descriptors contained the CPS-Packet header information and the absolute address of the CPS-Packet in the TX SSRAM.
- The VC Number in the CPS-Packet Final Assembly Structure points to the VC Send Event structure. A VC Send Event structure is only generated when enough payload is obtained to fill an ATM cell or the cell has past the expiry period. This is denoted by the Pending Payload Byte or Cell Expiry Time Stamp field in the AAL2 VC Structure.
- Upon a VC Send Event, the TX SAR writes a complete ATM cell (see Figure 21 on page 43) to the TX SAR Input FIFO (4 cell depth).

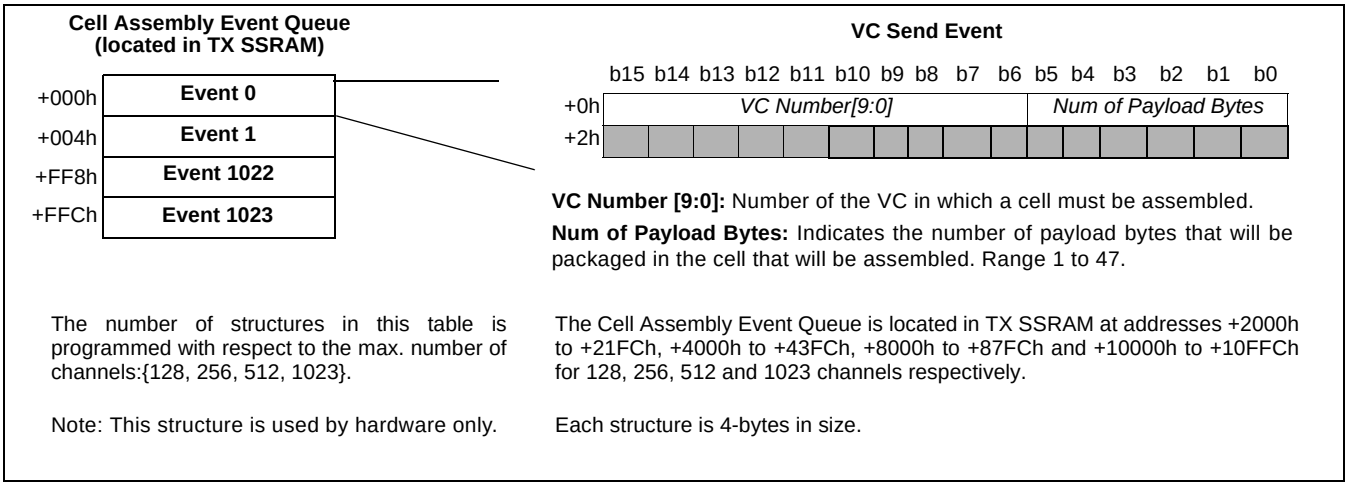



**Figure 20 - Cell Assembly Event Queue**

.

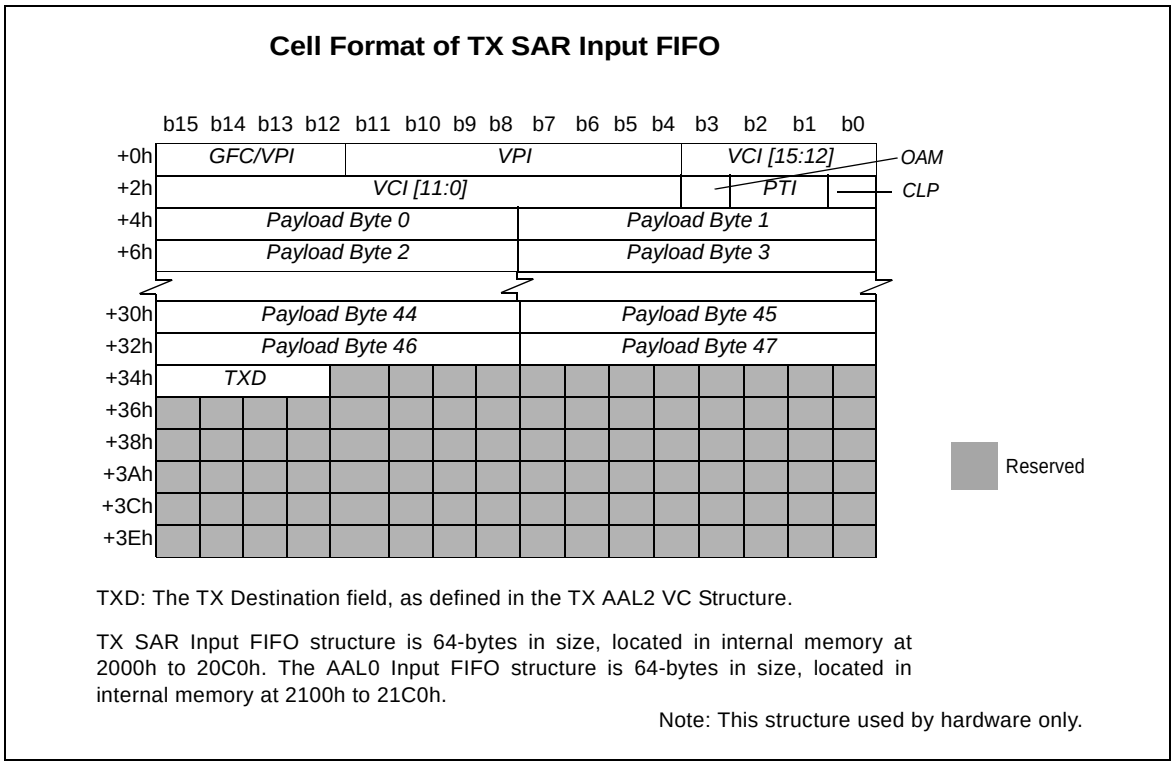



**Figure 21 - TX Cell Format**

### 2.3.3 AAL0 Cells

The TX SAR has the ability to insert AAL0 cells in the transmission direction. AAL0 cells are generated by the CPU and are directed to a 4-cell internal FIFO memory. It is necessary for the CPU to read the AAL0_MONITOR register (310h) to ascertain if the 4-cell internal FIFO memory can accommodated the pending AAL0 cell. The destination of the AAL0 cell is determined by the TX Destination field.

The aal0_input_fifo is 4 cells deep in the MT90502, however, it is the hardware that manages the pointers. The software will write the cell to the first entry (0x2100 to 0x213E) and then it will set the register indicating that there is a new cell. There is more than 1 cell to write at a time and the aal0_input_cell_written is set after each cell.

## 2.4 RX SAR

The RX SAR module performs processing on ATM cells received from the UTOPIA module. Cells placed in the RX SAR Output FIFO by the UTOPIA module are first read, then divided into the component CPS-Packets, and finally written into the appropriate CPS-Packet data circular buffer in external memory. The processing involves

- identifying the VC corresponding to the cell
- examining the cell for errors
- determining the routing of each CID, and
- disassembling the cell's CPS-Packets.

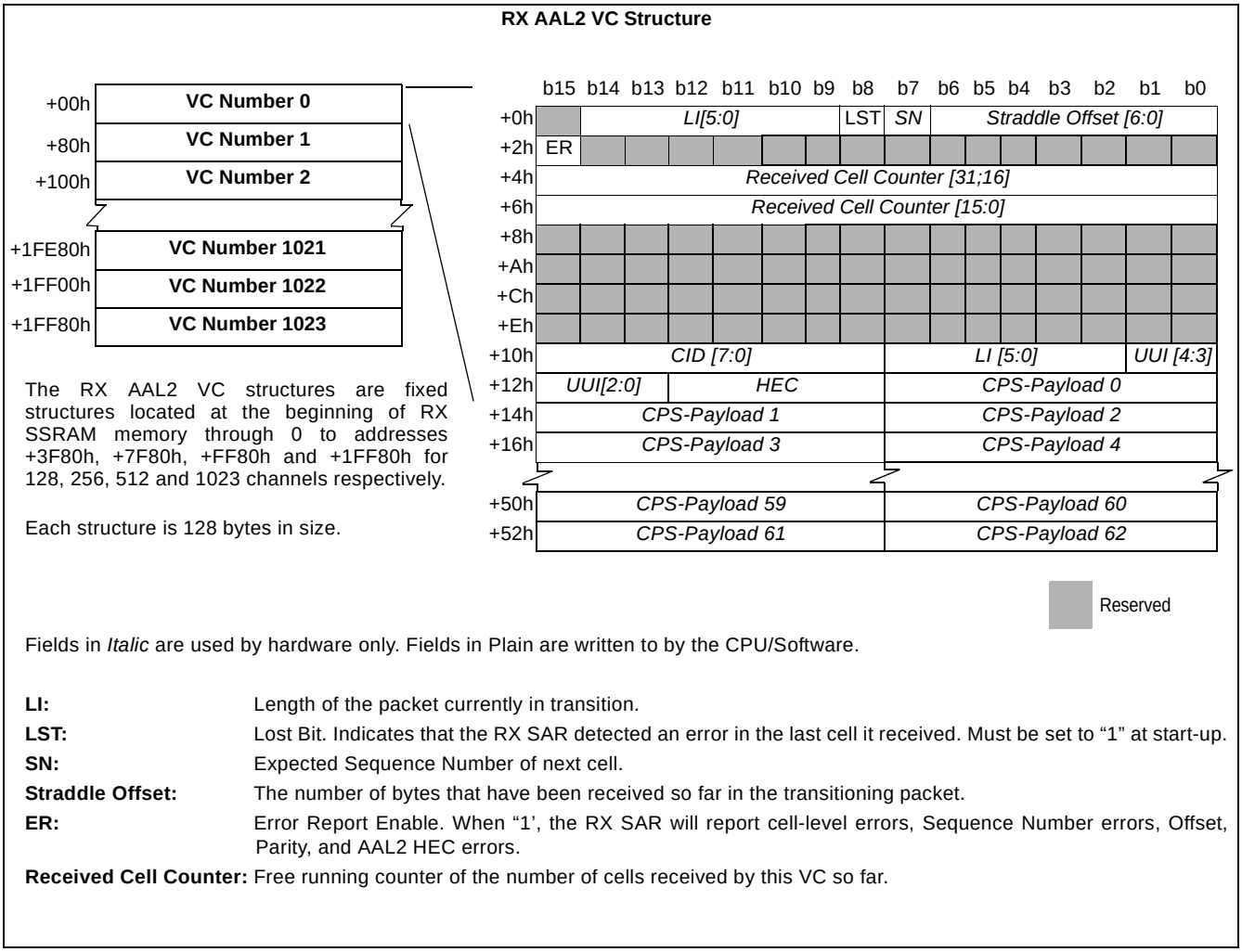
The status of the circular buffers is monitored for underruns and overruns. The RX SAR module also directs data cells to the data cell FIFO from which the CPU can read them. The RX SAR module supports PCM and ADPCM CPS-Packets of 1, 2, 3, 4, 5, or 8 EDUs, 32K ADPCM 80 frame CPS-Packets, 32K ADPCM 88 frame CPS-Packets, PCM 44 frame CPS-Packets, HDLC CPS-Packets of 1 to 64-bytes in length, and can accommodate packets straddling multiple cells.

### 2.4.1 RX AAL2 VC Structure

For each VC directed to the RX SAR, an RX AAL2 VC structure exists in external memory. The RX AAL2 VC structures contain the following:

- information required to detect errors
- a free-running received cell counter
- bytes received of from a CPS-Packet that is straddling the current and next AAL2 cells.

When a VC is opened, the lost bit must be set, the error report enable bit must be set (if per-VC errors are to be reported) or reset (if no per-VC errors are to be reported), and the received cell counter must be initialized to 0.




The RX AAL2 VC structures are fixed structures located at the beginning of RX SSRAM memory through 0 to addresses +3F80h, +7F80h, +FF80h and +1FF80h for 128, 256, 512 and 1023 channels respectively.

Each structure is 128 bytes in size.

Fields in *Italic* are used by hardware only. Fields in Plain are written to by the CPU/Software.

**LI:** Length of the packet currently in transition.

**LST:** Lost Bit. Indicates that the RX SAR detected an error in the last cell it received. Must be set to “1” at start-up.

**SN:** Expected Sequence Number of next cell.

**Straddle Offset:** The number of bytes that have been received so far in the transitioning packet.

**ER:** Error Report Enable. When “1’, the RX SAR will report cell-level errors, Sequence Number errors, Offset, Parity, and AAL2 HEC errors.

**Received Cell Counter:** Free running counter of the number of cells received by this VC so far.

**Figure 22 - RX AAL2 VC Structure**

### 2.4.2 CID Structure

Once cells are received by the RX SAR, they are broken down into their component CPS-Packets, and analysed on a per-CID basis. The RX SAR contains 255 RX CID descriptor structures per VC, each one indicating where CPS-Packets containing that CID should be routed. Each CPS-Packet can be treated either as a PCM channel, an HDLC channel, a CPU CPS-Packet, or it can be discarded entirely. This routing is based on the UUI of the CPS-Packet.

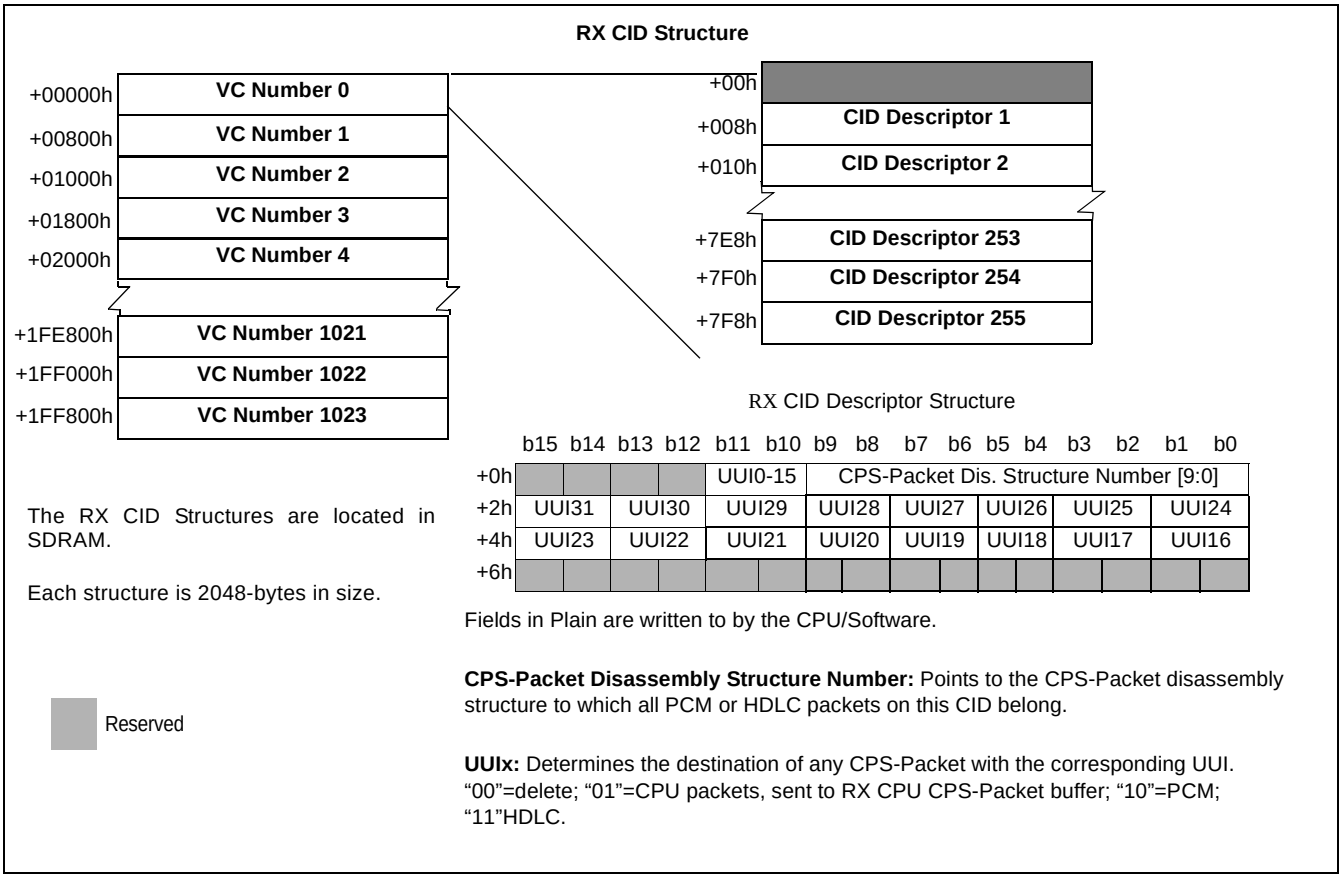



**Figure 23 - RX CID Structure**

The CPS-Packet Disassembly Structure field indicates for PCM and HDLC channels the CPS-Packet disassembly structure to be used for CPS-Packets identified by the CID Descriptor Structure. UUIs 0 to 15 are routed together because the 4 LSBs of the UUI can be used as a sequence counter. The remaining 16 UUIs are routed separately. Each one can be destined to the CPU, to the PCM or HDLC channel, or deleted entirely.

### 2.4.3 CPS-Packet Disassembly Structures

CPS-Packets routed to PCM or HDLC channels are treated by the CPS-Packet disassembly structure. Though the CPS-Packet disassembly structure is in the same location for PCM and HDLC channels, the fields are interpreted as one or the other depending on the value of the appropriate UUI field in the corresponding CID descriptor structure. The fields in the CPS-Packet disassembly structure indicate the form of the channel and the type of treatment required on the CPS-Packet to be disassembled.

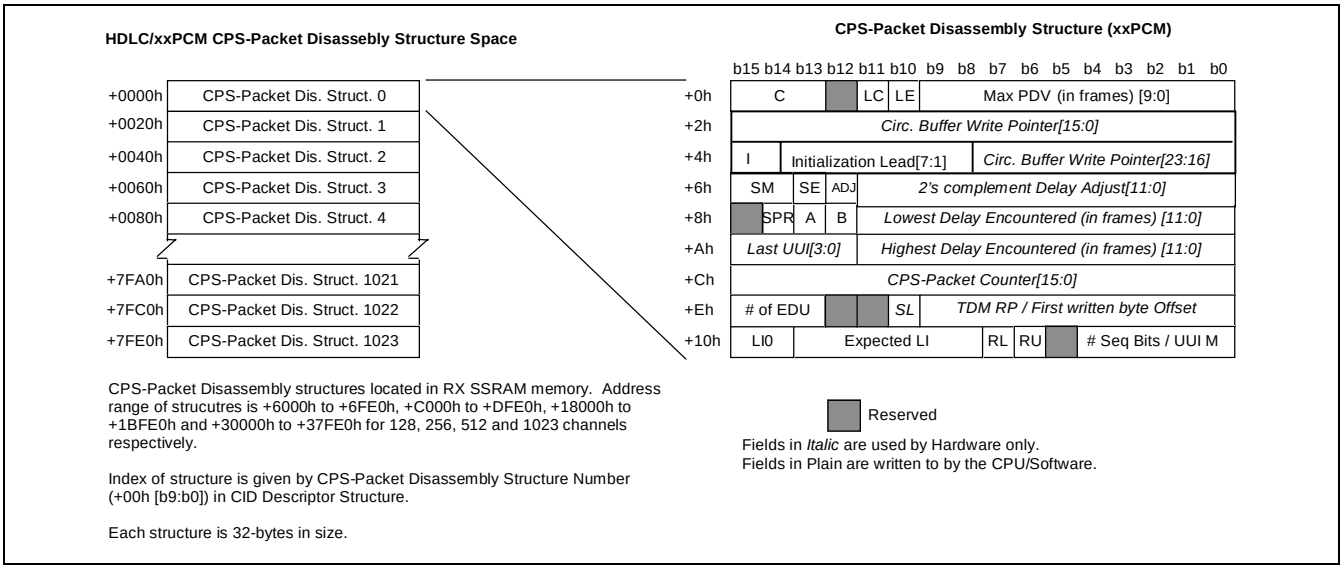



**Figure 24 - CPS-Packet Disassembly Structure (PCM/ADPCM)**

| Field | Name of Field | Byte Address Offset/Bits Used | Description of Field |
|---|---|---|---|
| C | Compression Ratio | +0/b15:b13 | '000' = 64 kbps, '010' = 40 kbps, '011' = 24 kbps, '100' = 16 kbps, '101' = ADPCM auto-detect (can change between 16, 24, 32 and 40 kbps), '110' = PCM & ADPCM auto-detect (can change between 16, 24, 32, 40, 64 kbps), others = reserved. |
| LC | CPS-Packet Loss Compensation Enable | +0/b11 | 0: CPS-Packet Loss Compensation circuit disabled.<br>1: CPS-Packet Loss Compensation circuit enabled. |
| LE | Loss Error Report Enable | +0/b10 | This bit must be set in order for CPS-Packet loss errors (e.g. losses due to silence suppression or to network quality) to be reported on this CID.<br>CPS-Packet loss errors are detected by circular buffer overflow, or due to CPS-Packet descriptor queue overrun (in HDLC), and by sequence number errors (in PCM). |
| Max PDV (in frames) [9:0] | Max PDV (in frames) | +0/b9:b0 | This indicates the maximum quantity of PDV that the AAL2 Channel will incur. It is measured in frames (i.e. 125 us). A value of 0 implies that there is no PDV on the AAL2 Channel. The maximum value for this field depends on the Number of EDUs and the size of the RX Circular Buffers. It is defined as ((Size of circular buffer) - 8*(Number of EDUs) -4) /2. |
| Circ. Buffer Write Pointer [23:0] | Circular buffer write pointer | +2/b15:b0, +4/b7:b0 | Pointer to the next memory location to be written upon the arrival of the next CPS-Packet. |
| I | Structure Initialized bit | +4/b15 | Structure initialized by hardware bit. Written to '0' by software upon opening the channel, written to '1' by hardware upon receiving the first CPS-Packet. |

**Table 22 - CPS-Packet Disassembly Structure (PCM/ADPCM) Fields**

| Field | Name of Field | Byte Address Offset/Bits Used | Description of Field |
|---|---|---|---|
| Initialisation Lead [7:1] | Initialisation Lead | +4/b14:b8 | Delay in frames that the first byte of the first CPS-Packet will go through before being sent on the TDM bus. This can be set to a lower value in order to minimise delay or to a higher value to avoid slips during the first CPS-Packets of the connection. Its value cannot exceed the Max PDV field. Typically, this field would be set to (Max PDV)/2 when initializing in the middle, or to 1 or 2 when initializing close to the edge. |
| SM | Slip Mode | +6/b15:b14 | Action to be taken upon a slip.<br>‘0x’ = Slip to edge of circular buffer on underruns<br>‘1x’ = Slip to middle of circular buffer on underruns<br>‘x0’ = Slip to edge of circular buffer on overruns<br>‘x1’ = Slip to middle of circular buffer on overruns |
| SE | Slip Error Report Enable | +6/b13 | When ‘0’, underrun and overrun slips will not be reported. When ‘1’, they will be reported. |
| ADJ | Delay Adjustment pending | +6/b12 | Written to ‘1’ by software when a delay adjustment must be done by the hardware. Hardware will clear this field once the delay adjustment has been performed. |
| 2’s complement Delay Adjust[11:0] | 2’s Complement Delay Adjust | +6/b11:b0 | Two’s complement number that will be added to the write pointer when the next CPS-Packet arrives. A value of 0 will not modify the delay. A value of -1 will reduce the delay by one frame. A value of +1 will increase the delay by one frame. Any time an adjustment is performed the Lowest and Highest delay fields will be adjusted accordingly. |
| SPR | Silence Suppression Enable | +8/b14 | When ‘1’, any byte matching the “silent_pattern” (contained in register 0410h) will not be written to the external memory, causing silent padding to be inserted instead. |
| A | Clock recovery reference A | +8/b13 | When ‘1’, a pulse will be sent to the clock recovery module indicating that a reference packet (a timing CPS-Packet) has been received. This bit corresponds to clock recovery channel A which is independent of clock recovery channel B. |
| B | Clock recovery reference B | +8/b12 | When ‘1’ a pulse will be sent to the clock recovery module indicating that a reference packet (a timing CPS-Packet) has been received. This bit corresponds to clock recovery channel B which is independent of clock recovery channel A. |
| Lowest Delay Encountered (in frames) [11:0] | Lowest delay encountered | +8/b11:b0 | Lowest delay inserted by the hardware’s buffer. This is a signed number: 0 is a valid delay of less than one frame,<br>-1 is an invalid delay and indicates that one or more slips have occurred. |
| Last UUI [3:0] | Last UUI | +A/b15:b12 | Last received UUI bits 3:0. |

**Table 22 - CPS-Packet Disassembly Structure (PCM/ADPCM) Fields (continued)**

| Field | Name of Field | Byte Address Offset/Bits Used | Description of Field |
|---|---|---|---|
| Highest Delay Encountered (in frames) [11:0] | Highest delay encountered | +A/b11:b0 | Highest delay inserted by the hardware's buffers. This is a signed number. All delays up to Max PDV in Frames are acceptable. Beyond this point, one or more slips have occurred. |
| CPS-Packet Counter[15:0] | CPS-Packet counter | +C/b15:b0 | Free running count of received CPS-Packets (SID and non-SID included). |
| # of EDU | Number of EDU | +E/b15:b13 | Number of data units in each packet.<br>'000' = 8 frames or 1 EDU<br>'001' = 16 frames or 2 EDU<br>'010' = 24 frames or 3EDU<br>'011' = 32 frames or 4EDU<br>'100' = 40 frames or 5 EDU<br>'101' = 44/88 frames PCM/ADPCM32<br>'110' = 40/80 frames PCM/ADPCM32<br>'111' = 64 frames or 8 EDU |
| SL | SID Last | +E/b10 | When '1', the last packet received was a SID. This field is used to generate the ADPCM reset if the first voice packet after a SID contains ADPCM data. This bit should be initialized to '1' so that the first ADPCM packet resets the decoder. |
| TDM RP / First written byte Offset | TDM RP/First written byte offset | +E/b9:b0 | Positive offset between the circular buffer write pointer used to write the first byte of a CPS-Packet and the TDM_circular_buffer_read_pointer + 1. initialize to "00_0000_0000" |
| LI0 | Length indicator of zero | +10/b15:b14 | Indicates how to treat CPS-Packets with the LI field equal to 0.<br>'00' = reserved.<br>'01' = CPS-Packet is SID, discard CPS-Packet.<br>'10' = CPS-Packet is SID, treat CPS-Packet.<br>'11' = CPS-Packet is SID, treat CPS-Packet and generate report structure. |
| Expected LI | Expected LI | +10/b13:b8 | The LI expected in all voice packets. If the received LI does not match the Expected LI and RL = '1', then an error report structure will be generated. Note: the CPS-Packet will still be treated. |
| RL | Report Length Errors | +10/b7 | Report each CPS-Packet received in which the LI does not match Expected LI. |
| RU | Report UUI Errors | +10/b6 | Report each CPS-Packet received in which the UUI does not match expected UUI. |
| # Seq Bits/UUI M | Number of Sequence Bits in the UUI and UUI Match | +10/b4:b0 | '10000' = 4 sequence bits<br>'x1000' = 3 sequence bits & UUI[3] in match<br>'xx100' = 2 sequence bits & UUI[3:2] in match<br>'xxx10' = 1 sequence bit & UUI[3:1] in match<br>'xxxx1' = 0 sequence bits & UUI[3:0] in match |

**Table 22 - CPS-Packet Disassembly Structure (PCM/ADPCM) Fields (continued)**

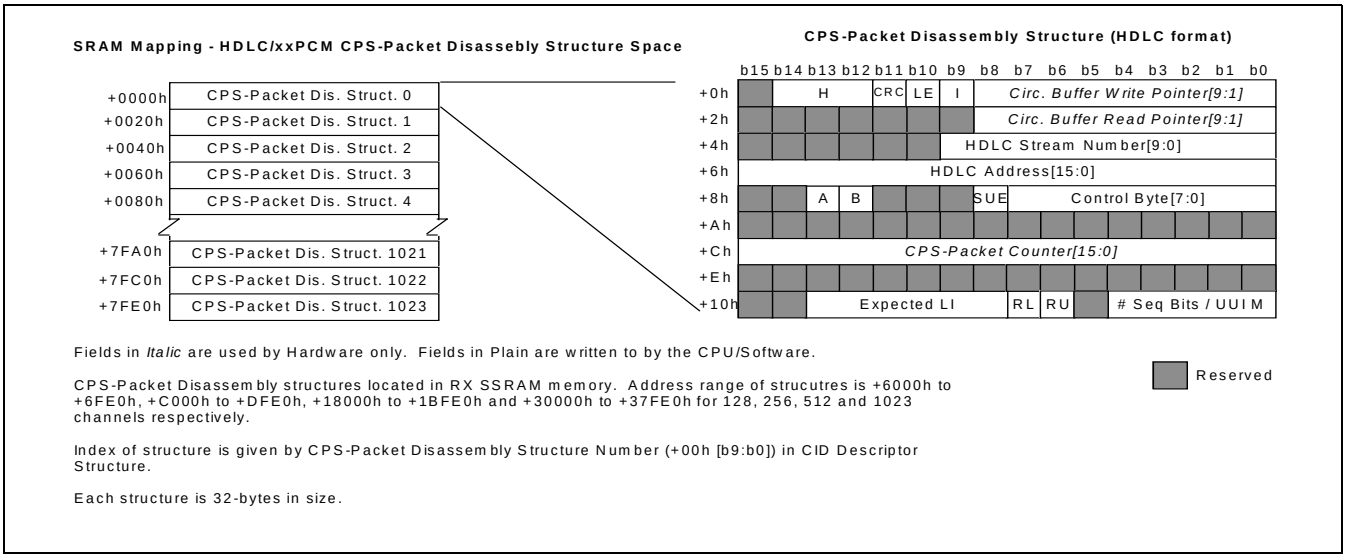



**Figure 25 - CPS-Packet Disassembly Structure (HDLC Format)**

| Field | Name of Field | Byte Address Offset/Bits Used | Description of Field |
|---|---|---|---|
| H | Header Type | +0/b14:b12 | Gives the format of the HDLC header<br>‘000’ = no header bytes<br>‘001’ = one byte address<br>‘010’ = two byte address<br>‘011’ = one byte address + control<br>‘100’ = two byte address + control<br>others = reserved |
| CRC | CRC enable | +0/b11 | Indicates if a 16-bit trailing CRC is to be appended to the CPS-Packet. |
| LE | Loss Error Report Enable | +0/b10 | This bit must be set in order for CPS-Packet loss errors (e.g. losses due to silence suppression or to network quality) to be reported on this CID. CPS-Packet loss errors are detected by circular buffer overflow, or due to CPS-Packet descriptor queue overrun (in HDLC), and by sequence number errors (in PCM). |
| I | Structure Initialized bit | +0/b9 | Reset by software upon opening the channel, set by hardware upon receiving the first CPS-Packet. |
| Circ. Buffer Write Pointer [9:1] | Circular Buffer Write Pointer | +0/b8:b0 | Points to the location in the circular buffer used to contain HDLC packets where the next CPS-Packet will be written. |
| Circ. Buffer Read Pointer [9:1] | Circular Buffer Read Pointer | +2/b8:b0 | Points to the location in the circular buffer used to contain HDLC packets where the next byte is to be read to be sent on the TDM bus |
| HDLC Stream Number [9:0] | HDLC Stream Number | +4/b9:b0 | Pointer to beginning of the HDLC Stream structure in the RX_TDM control memory to which this channel must be routed. Directing many HDLC streams to the same HDLC Stream structure allows many HDLC channels to point to a single stream. |

**Table 23 - CPS-Packet Disassembly Structure (HDLC format) Fields**

| Field | Name of Field | Byte Address Offset/Bits Used | Description of Field |
|---|---|---|---|
| HDLC Address[15:0] | HDLC Address [15:0] | +6/b15:b0 | Address used to form the HDLC header. This field is ignored if the Header Type is ‘000’ |
| A | Clock recovery reference A | +8/b13 | When ‘1’, a pulse will be sent to the clock recovery module indicating that a reference packet (a timing CPS-Packet) has been received. This bit corresponds to clock recovery channel A which is independent of clock recovery channel B. |
| B | Clock recovery reference B | +8/b12 | When ‘1’, a pulse will be sent to the clock recovery module indicating that a reference packet (a timing CPS-Packet) has been received. This bit corresponds to clock recovery channel B which is independent of clock recovery channel A. |
| SUE | Send UUI enable | +8/b8 | Send UUI in control byte enable. When set and when Header Type is ‘011’ or ‘100’, the 5 MSBs of the HDLC control byte will be the UUI. |
| Control Byte [7:0] | HDLC Control Byte | +8/b7:b0 | Control byte used to form the HDLC control byte. This field is ignored if the Header Type is not ‘011’ or ‘100’. |
| CPS-Packet Counter[15:0] | CPS-Packet counter | +C/b15:b0 | Free running count of received CPS-Packets (SID included). |
| Expected LI | Expected LI | +10/b13:b8 | The LI expected in all voice packets. If the received LI does not match the Expected LI and RL = ‘1’, then an error report structure will be generated. Note: the CPS-Packet will still be treated. |
| RL | Report Length Errors | +10/b7 | Report each CPS-Packet received in which the LI does not match Expected LI. |
| RU | Report UUI Errors | +10/b6 | Report each CPS-Packet received in which the UUI does not match expected UUI. |
| # Seq Bits/UUI M | Number Sequence Bits in the UUI and UUI Match | +10/b4:b0 | ‘10000’ = 4 sequence bits<br>‘x1000’ = 3 sequence bits & UUI[3] in match<br>‘xx100’ = 2 sequence bits & UUI[3:2] in match<br>‘xxx10’ = 1 sequence bit & UUI[3:1] in match<br>‘xxxx1’ = 0 sequence bits & UUI[3:0] in match |

**Table 23 - CPS-Packet Disassembly Structure (HDLC format) Fields (continued)**

**Note:** A and B bits in both the PCM and HDLC CPS-Packet disassembly structures are used to signal the chip when a timing reference CPS-Packet has been received. As the UTOPIA module can also generate these signals based on timing reference cells, each of these bits should be set in only one location: either in a single LUT or in a single CPS-Packet disassembly structure.

The compression rate of the ADPCM CPS-Packets can be dynamically determined using the EDU field and the LI field of the CPS-Packet. Once the compression rate is determined for a CID, the ADPCM samples will be formatted according to the compression rate (see Figure 26 on page 52) and placed in the Rx Circular Buffer.

To alter the monitored delay, software can generate a delay adjustment by writing the ADJ field to ‘1’ and the 2’s Complement Delay Adjust field to the number of bytes by which the delay should be altered.

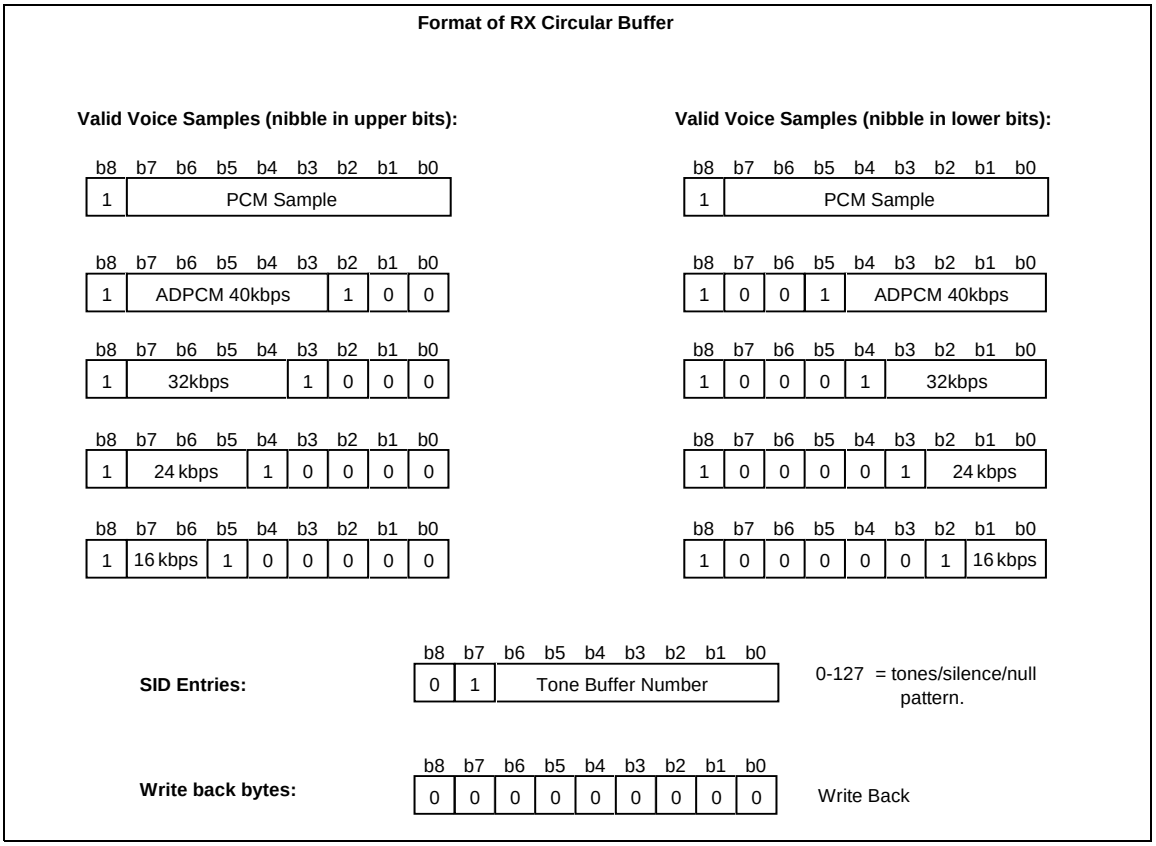



**Figure 26 - Format of RX Circular Buffer**

For HDLC CPS-Packets, the RX SAR performs zero-insertion and appends the HDLC header and CRC (as necessary) to the packet before writing it to the circular buffer. The RX SAR can be configured to either bit-wise or byte-wise zero-insertion (one method for all of the HDLC channels). It can also write packets either with or without an AAL2 header.

### 2.4.4 CPS-Packet Loss Compensation

The MT90502 is capable of calculating the elapsed time between the reception of two CPS-Packets. Based on the time elapsed and the difference in the sequence number in the UUI of the received CPS-Packet and that of the previous CPS-Packet, a count of how many CPS-Packets have been lost can be found. The circular buffer write pointer is adjusted to take the number of CPS-Packets lost into account before beginning to write the received bytes. This algorithm is only successful if the PDV between packets is smaller than a complete wrap of the UUI sequence counter. An enable bit (LC) in the CPS-Packet disassembly structure allows the CPS-Packet loss compensation to be turned on or off.

### 2.4.5 CPU CPS-Packets

If the CID Description Structure denotes a CPS-Packet as a CPU CPS-Packet, the CPS-Packet will be written into the CPU CPS-Packet FIFO in external memory and a CPU CPS-Packet report structure will be generated (see Section 2.4.7, Errors and Events on page 53). The CPU is alerted to the presence of CPU CPS-Packets through the CPU CPS-Packet report structures. The CPU CPS-Packet base address can be read from the CPS-Packet report structure.

The size of the CPU CPS-Packet FIFO is programmable (register 440h) as is its base address (register 440h). The size of the FIFO can vary from 16 KB to 128 KB. The CPU is responsible for updating the read pointer (register 442h) following each read.

### 2.4.6 Treatment of Data Cells

Data cells, such as those containing OAM information, are placed in a programmable length FIFO in external memory. The length of the FIFO is stored in register 430h. The CPU can read one or more data cells from the FIFO at any time, after obtaining the address of the FIFO (register 430h) and the read pointer (register 432h). The CPU is responsible for updating the read pointer following each read. The write pointer, incremented by hardware, indicates the location where the next data cell will be written. The FIFO is empty is the read pointer points to the same location as the write pointer.

The CPU can be alerted to the presence of data cells via an interrupt that triggers if either of two events occur:

1. the interrupt can be generated when the FIFO becomes more than half full, or
2. the interrupt can be generated if a data cell has been present in the FIFO for longer than a programmable period of time (registers 460h, 462h).

When ready to process the information, the CPU obtains a read pointer to the information from register 432h and reads the information through 26-word accesses.

Cells with the OAM bit set in the PTI portion of the header can be directed to the data cell FIFO on a per VC basis. The same is true for non-OAM cells. In addition, unknown non-OAM cells and/or unknown OAM cells can also be sent to the data cell FIFO. All unknown non-OAM cells are directed to the same location(s), and all unknown OAM cells are directed to the same location(s).

### 2.4.7 Errors and Events

An Error/Event Report FIFO exists in RX SSRAM whose base address and size are configured through register 438h. Four types of structures, as shown in Figure 26, are available to cover all errors or events listed in Table 24. Once an error or event occurs, a corresponding 4-word structure will be generated and written into Report FIFO. The error/event structure contains all the details pertaining to that error or event.

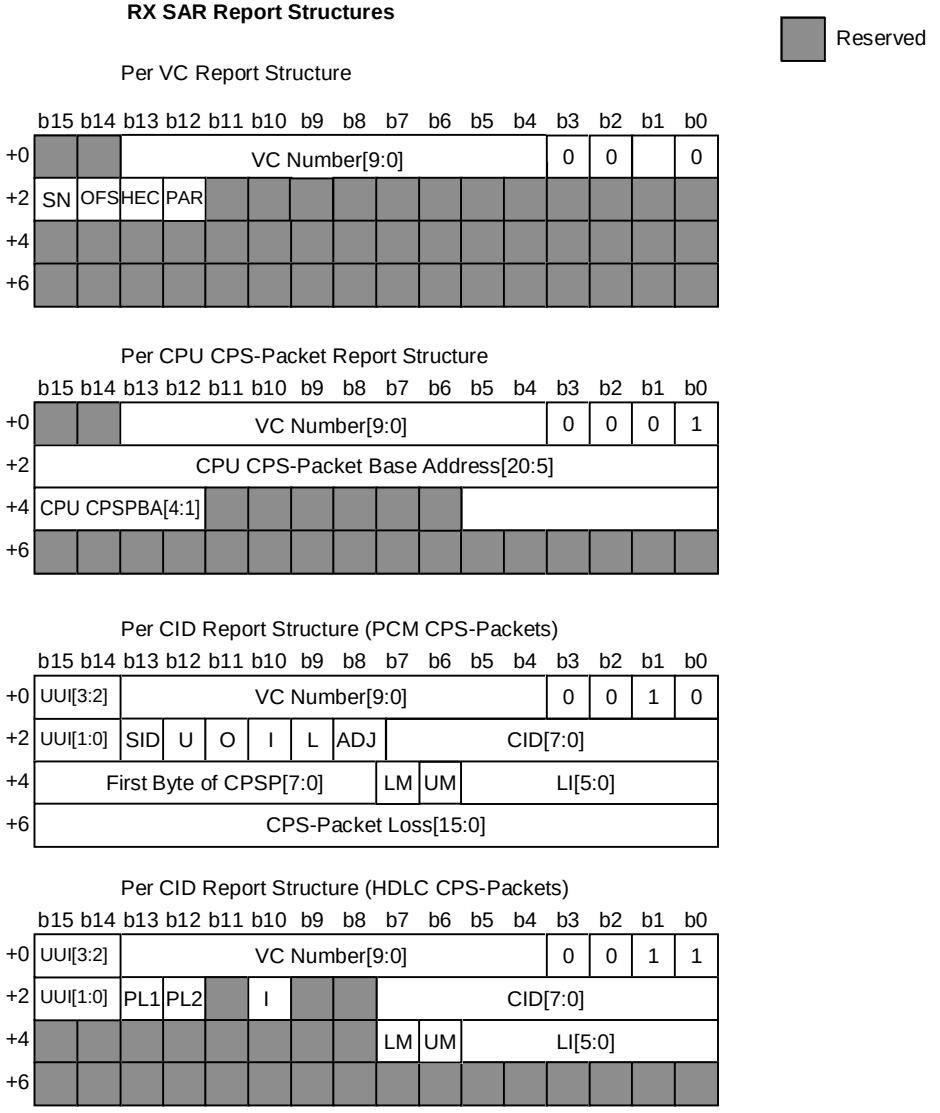
Similar to the data cell FIFO, the CPU can read the error/event FIFO at any time after obtaining the base address (register 438h) and the read pointer of the error/event FIFO (register 43Ch). Again, an interrupt can be generated that triggers if either of two events occur. Those events are

- the FIFO becomes more than half full, or
- an error/event report structure has been present in the FIFO for longer than a programmable period of time (registers 464h, 466h).

When ready to process the information, the CPU obtains a read pointer to the information from register 43Ch and reads the information through 4-word accesses.

**Note:** Both PCM and HDLC CPS-Packets will generate a report structure when they are initialized. Apart from the generation of the error report structure, no action is taken on errors/events.

**RX SAR Report Structures**




**VC Number**: Number of the VC on which the error occured.
**SN**: Sequence Number Error.
**OFS**: Offset Error.
**HEC**: AAL2 CPS-Packet Header HEC Error.
**PAR**: Offset Field parity error.
**PL1**: Complete Packet Lost because of data buffer overflow (HDLC only).
**PL2**: Complete Packet Lost because of CPSPDQ overflow (HDLC only).
**SID**: SID CPS-packet received.
**U**: Underrun Detected. (xxPCM only)
**O**: Overrun Detected. (xxPCM only)
**I**: CID Initialized. First CPS-Packet has been received. (Both HDLC and PCM)
**L**: Loss of one or many CPS-Packet detected. See CPS-Packet Loss for a count of loss packets. (xxPCM only)
**ADJ**: CDV Adjustment Performed. (xxPCM only)
**LM**: LI Mismatch.
**UM**: UUI Mismatch.
**First Byte of CPS-Packet**: This is the first byte of the CPS-Packet. When SID = '1', this is the SID byte.

**Figure 27 - RX Error Report FIFO Structures**

| Category | Enable Location | Error/Event | Coverage |
|---|---|---|---|
| Cell-specific (per VC) | ER bit in each RX SAR disassembly structure | Sequence number error | AAL2 cells |
| | | Offset Field error | AAL2 cells |
| | | ATM HEC error | all ATM cells |
| | | Offset Field parity error | AAL2 cells |
| CPU CPS-Packet | none | Reception of a CPU CPS-Packet | CPU directed CPS-Packets |
| PCM CPS-Packets (per CID) | LE in PCM CPS-Packet disassembly structure | SID CPS-Packet received | PCM CPS-Packets |
| | | Underrun error | PCM CPS-Packets |
| | | Overrun error | PCM CPS-Packets |
| | | CID initialized | PCM CPS-Packets |
| | | Loss of one or more CPS-Packets error | PCM CPS-Packets |
| | | PDV adjustment performed | PCM CPS-Packets |
| | | LI mismatch error | PCM CPS-Packets |
| HDLC CPS-Packets (per CID) | LE in HDLC CPS-Packet disassembly structure | data buffer overflow error | HDLC CPS-Packets |
| | | CPS-Packet disassembly queue overflow error | HDLC CPS-Packets |
| | | CID initialized | HDLC CPS-Packets |
| | | LI mismatch error | HDLC CPS-Packets |

**Table 24 - RX SAR Errors and Events**

## 2.5 TDM Reception

### 2.5.1 Overview

The RX TDM Module is responsible for reading bytes written to circular buffers in external memory by the RX SAR and administering the data to the H.100/H.110 interface module (see Section 2.7, “H.100/H.110 Interface,” on page 70). The RX TDM Module is capable of transmitting data in 1023 time slots on the H.100/H.110 bus.

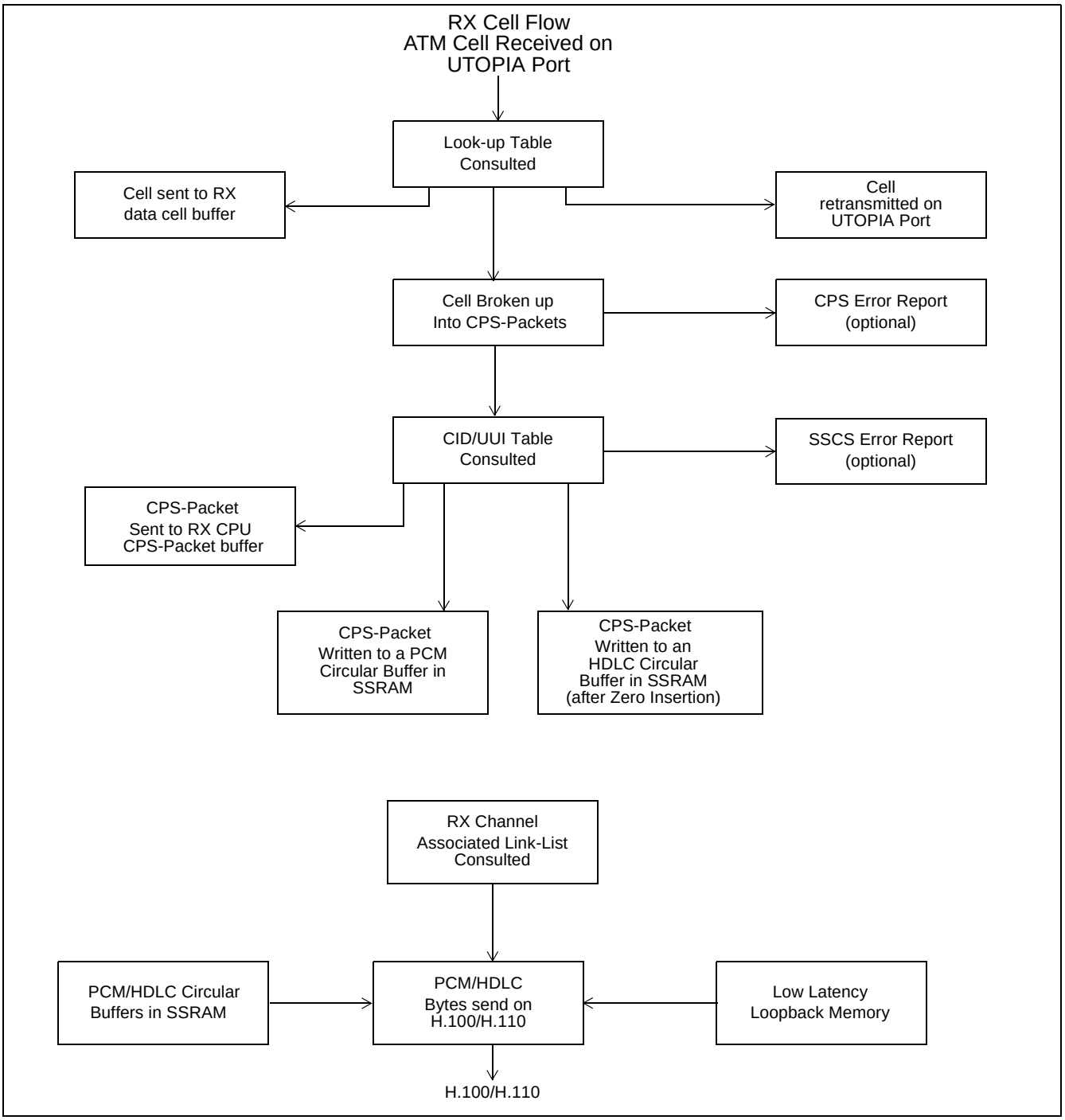



**Figure 28 - RX Cell Flow**

### 2.5.2 RX Channel Association Memory

The RX CAM generates a map that indicates which ADPCM channel, PCM channel or HDLC stream TSSTs are associated on the TDM bus. There are 1024 entries in RX CAM; the first entry is the permanent start entry for the map thus does not point to any TSST. The remaining 1023 entries are usable, each of which consists of

- the PCM channel/ADPCM channel/HDLC stream number, or Loopback channel number.
- the TSST number
- the pointer to the next mapped entry. Note that the final entry in the map must always point back to the start entry.

Like the TX CAM, all entries in the RX CAM must be linked sequentially in TSST order.

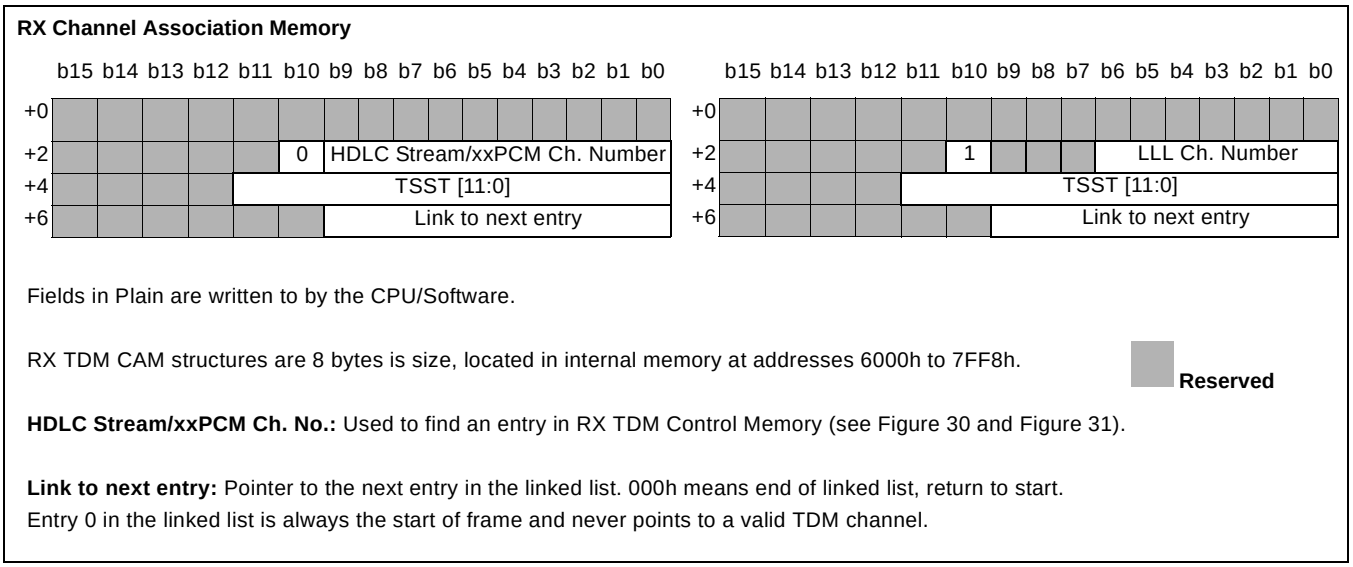



**Figure 29 - RX Channel Association Memory (RX CAM)**

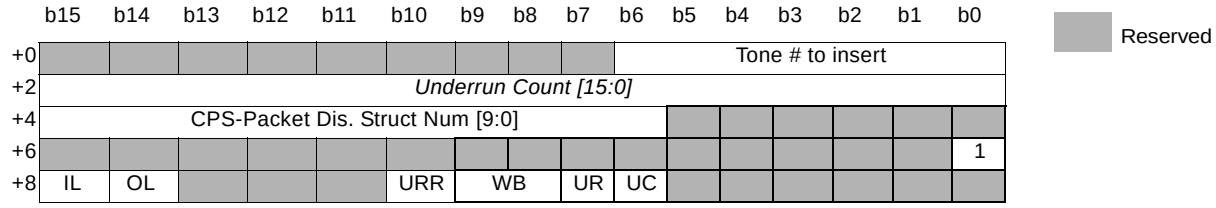
### 2.5.3 RX Channel Underrun Condition

When servicing PCM/ADPCM channels, a byte is read every frame from the external circular buffers and transmitted onto the TDM bus. PCM/ADPCM channels can be configured to perform write backs to the circular buffers after the read has been completed. This will ensure erroneous data will not be transmitted onto the TDM bus while an underrun condition occurs. The **WB** (**W**rite **B**ack type) bits in the RX Control Memory determine what is written back: tone/silent pattern (programmable in silent_pattern_reg 410h) or no Write Back. The PCM/ADPCM entry in RX Control Memory contains several fields that control underrun detection:

- 16-bit free-running underrun counter which counts the number of underruns that have occurred on this channel
- The **UC** (**U**nderrun **C**ount enable) bit enables the counter. The PCM/ADPCM control structure also contains the number of the PCM/ADPCM channel that is managed by the control entry.

For underrun detection to work properly, all RX CPS-packet circular buffer contents, including parity bit, must be initialized to zero.

RX TDM Control Memory Structure (PCM/ADPCM channels)




Fields in *Italic* are used by Hardware only.

Fields in Plain are written to by the CPU/Software.

RX TDM Control Memory structures for XXPCM Channels are 16-bytes in size and are located in internal memory at addresses C000h and FFF0h.

**Tone # to insert:** Indicates which tone/silence should be inserted when underruns occur. This can be used to generate DTMF, Dial-tones, Busy Tones, etc. 0-63 = tones in SSRAM; 64-95 = silent noises in SDRAM; 96-127 = insert programmable null byte. This number will be overwritten by SID packet.

**Underrun Count:** Free running counter of the number of underruns detected on this channel. This field is in units of one sample being lost due to CPS-Packet loss or underrun situations.

**CPS-Packet Disassembly**

**Structure Number:** Pointer to the AAL2 channel associated with this TDM channel. This field is used to locate the RX Circular Buffer in SSRAM.

IL: Input law (from received CPS-Packet). ‘0’ = u-law; ‘1’ = A-law

OL: Output law (sent on TDM bus). ‘0’ = u-law; ‘1’ = A-law. If IL and OL are equal, bytes will not be modified. Otherwise u-law/A-law conversion will be done.

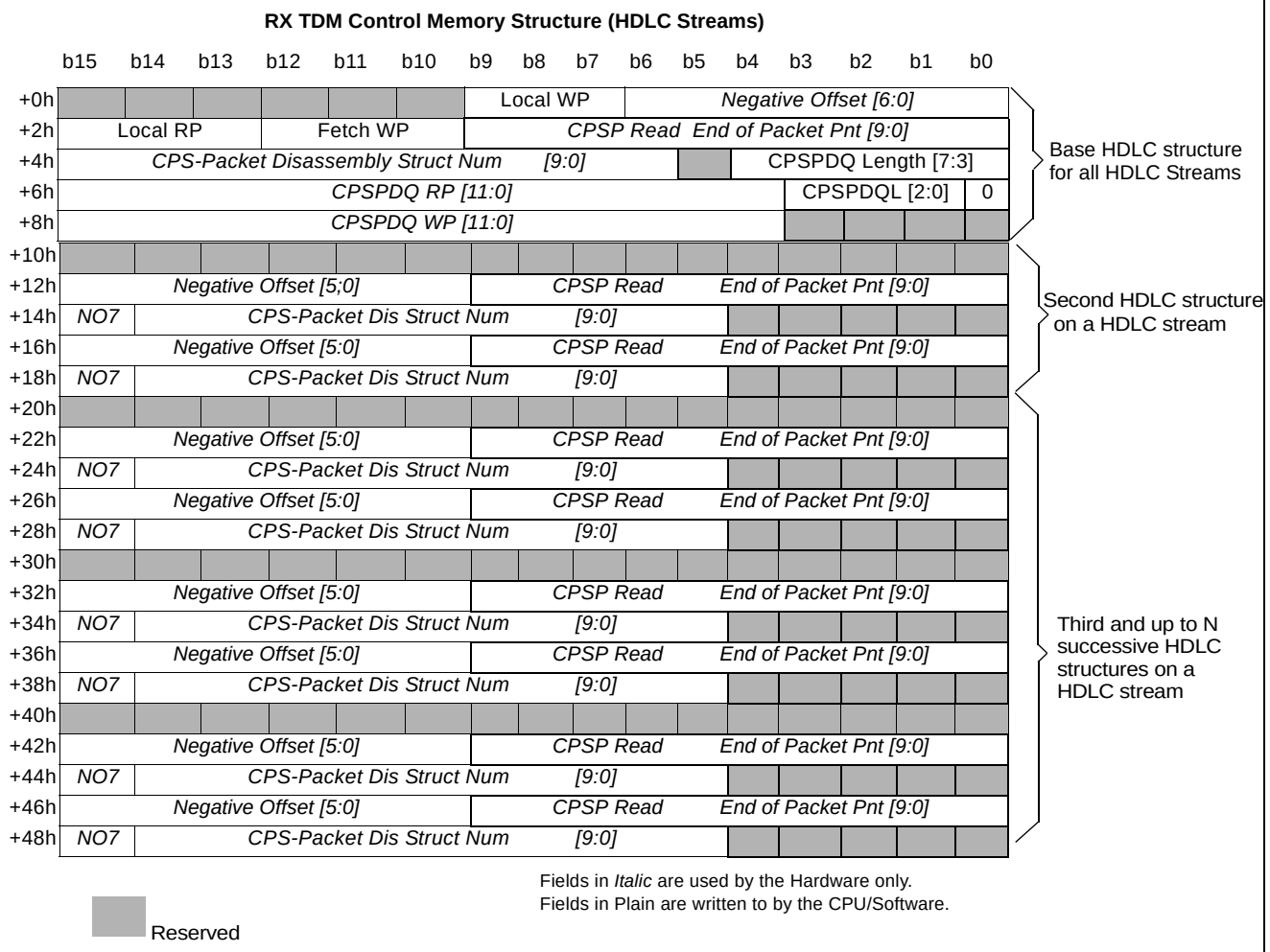
URR: Underrun Counter Rollover Report Enable.

WB: Write Back Type. ‘00’ = do not write back (replay buffer in case of underrun); ‘01’ = Write-back silent pattern in register 410h; ‘1x’ = reserved

UR: Underrun Report Enable.

UC: Underrun Count Enable (when ‘0’, the Underrun Counter is disabled; this saves interval bandwidth to the RX TDM Control memory).

**Figure 30 - RX TDM Control Memory Structure (PCM/ADPCM channels)**




RX TDM Control Memory structures for HDLC Streams are located in internal memory at addresses C000h to FFF0h.

**CPS-Packet Disassembly**

**Structure Number:** Pointer to the AAL2 channel associated with this HDLC stream. This field is used to locate the RX Circular Buffer in SSRAM.

**CPSPDQ Length:** Packet Descriptor Queue Length. Set to the number of timeslots in the stream. 01h = 1 TSST, 02h = 2 TSST, Max size is 128.

**Local WP/RP and Fetch WP:** Pointer to the pre-fetched HDLC packet handles contained in the 2/3/4/channel add on and the 5+ channel add-on. They must be initialized to 0h by software on HDLC stream start-up.

**CPSPDQ RP/WP:** Read/write pointer to this HDLC stream's Packet Descriptor Queue (PDQ) in SSRAM.

**CPSP Read End of Packet Pnt:** Pointer to the last octet in HDLC packet in the RX Circular Buffer.

**Negative Offset:** Current negative offset from the end of packet pointer at which the next byte of the HDLC packet will be read.

**Figure 31 - RX TDM Control Memory Structure (HDLC Streams)**

### 2.5.4 Compression

The RX TDM supports compression on PCM/ADPCM channels at the following data rates: 40, 32, 24 and 16 kbps. Each frame maintains a 5-, 4-, 3- or 2-bit compressed value in the high or low bits of the TDM byte and the remainder indicates the data rate (see Figure 7 on page 26 and Figure 10 on page 28).

### 2.5.5 HDLC

When processing HDLC channels on a per frame basis, the TDM RX Module will transmit either data bytes or HDLC null patterns (All 1s in bit-wise HDLC or 7Eh in byte-wise HDLC). See Section 2.10, HDLC on page 94. HDLC null patterns are transmitted between HDLC packets. The RX SAR communicates packet information to the RX TDM through a packet descriptor queue held in external memory. A maximum of 16 descriptors per channel may be retained in external memory. The descriptor contains the origin channel number of the AAL2 channel, base address, and length of the CPS-Packet within the buffer.

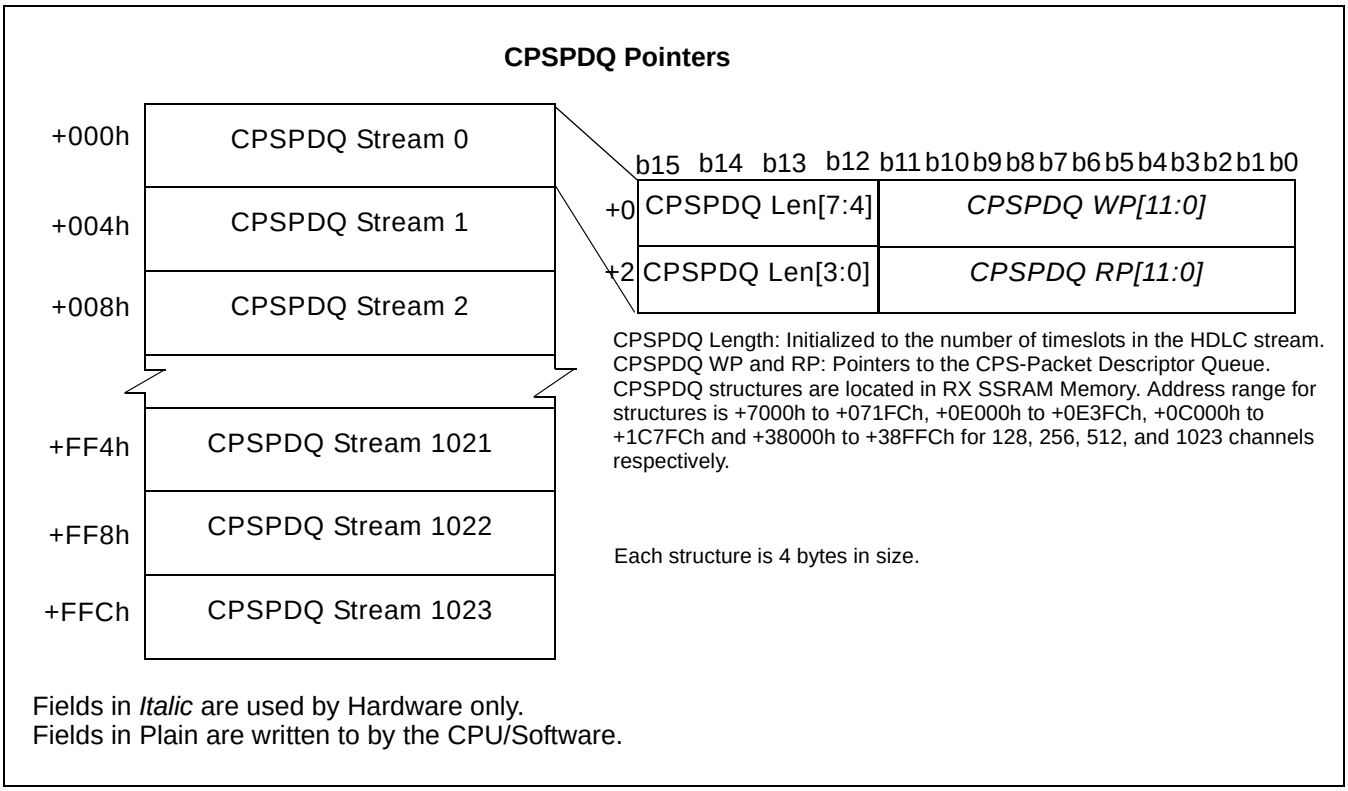
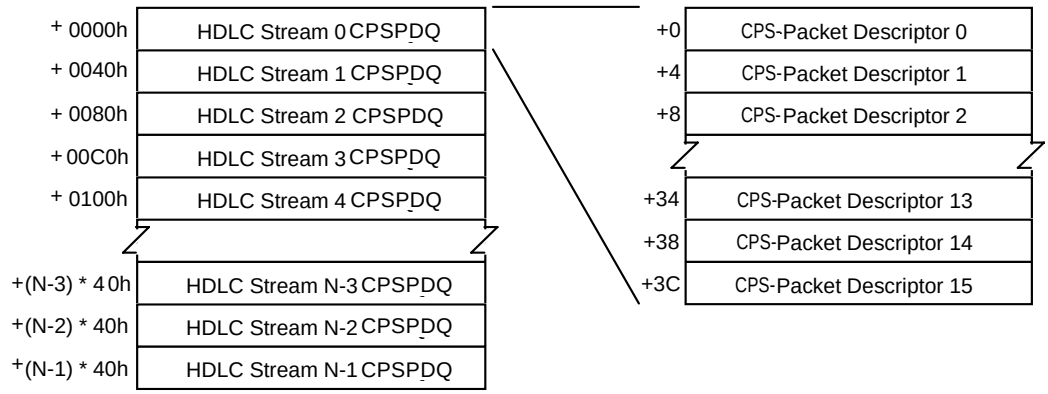



**Figure 32 - CPS-Packet Descriptor Queue Pointers Structure (HDLC Streams)**

**CPS-Packet Descriptor Queue Structures**

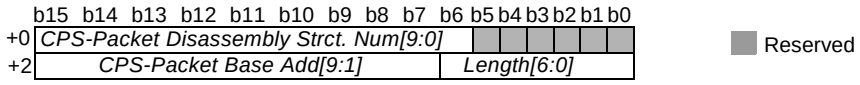



Number of HDLC Streams (sar capacity) is N = 128, 256, 512, 1023.

CPSPDQ structures are 64-bytes in size located in SSRAM memory at addresses +4000h to +5FC0h, +8000h to +BFC0h, +10000h to +17FC0h and +20000h to +2FFC0h for 128, 256, 512 and 1023 channels respectively.

This structure is used by Hardware only.

**CPS-Packet Descriptor Structure**




**CPS-Packet Disassembly Struct. Num** : Indicates which CPS-Packet disassembly structure this CPS-Packet comes from. This will identify the circular buffer in which it is written.

**CPS-Packet Base Add** : Points to the first byte of the CPS-Packet within the circular buffer.

**Length**: Length of the CPS-Packet, in bytes. Valid range 01h (1-byte) to 40h (64-bytes).

**Figure 33 - CPS-Packet Descriptor Queue Structures (HDLC Streams)**

As HDLC supports streams, numerous channel packet descriptor queues can be amalgamated into one packet descriptor per stream. However, it is necessary to specify the length of the packet descriptor queue within the structure that manages the stream; therefore, the MT90502 can determine where to wrap its read and write pointers. The length of the packet descriptor queue is programmed by the CPU/Software in the HDLC control structure (see Figure 30, “RX TDM Control Memory Structure (PCM/ADPCM channels),” on page 58). The CPS-Packet disassembly structures in the RX SAR determine where the HDLC channels are destined.

In HDLC, time slot entries in the linked list can point to the same control structure entry, allowing the HDLC single-channel or stream to span consecutive time slots within the same TDM stream. This is employed for HDLC streams where the bandwidth requirements often exceed the constraints of a single time slot. Since multiple entries pointing to the same HDLC channel must all be contained within the same TDM stream, the maximum number of time slots that can be shared by the same HDLC channel is 128.

## 2.6 UTOPIA

### 2.6.1 Overview

The purpose of the UTOPIA module is to provide an external interface with the ATM domain. The MT90502 complies with the ATM Forum's specifications: af-phy-0017.000 (PHY & SAR) and af-phy-0039.000 (PHY).

The UTOPIA module is responsible for accepting cells from five input interfaces: UTOPIA A, UTOPIA B, UTOPIA C, CPU Origin Data Cell FIFO, and TX SAR (internal). Cells are examined, and based on the origin and information in the header, the cell is sent to one or more (multicast or broadcast) of the four output interfaces: UTOPIA A, UTOPIA B, UTOPIA C or RX SAR (internal). The UTOPIA module employs octet level handshaking. It also appends the HEC to outgoing ATM cells.

The UTOPIA module has 2 different configurations and is constructed from three ports, labelled A, B, and C.

- Each UTOPIA port (A, B or C) can be independently configured as a 8-bit Level 1 PHY or ATM port, and can operate at up to 32MHz. In this configuration, ports can be employed to daisy-chain other SAR devices to the MT90502, or operate in a ring configuration. Each port can treat 155Mbps of bandwidth.
- Ports A and B can also be combined to architect one UTOPIA Level 2 port in PHY mode only. It has an 8-bit data bus and a 5-bit address bus. This configuration allows the MT90502 to interface with any UTOPIA Level 2 master device in a multiple-PHY application.

In addition to ports A, B, and C, the UTOPIA can route data cells to the cell FIFO in external SSRAM.

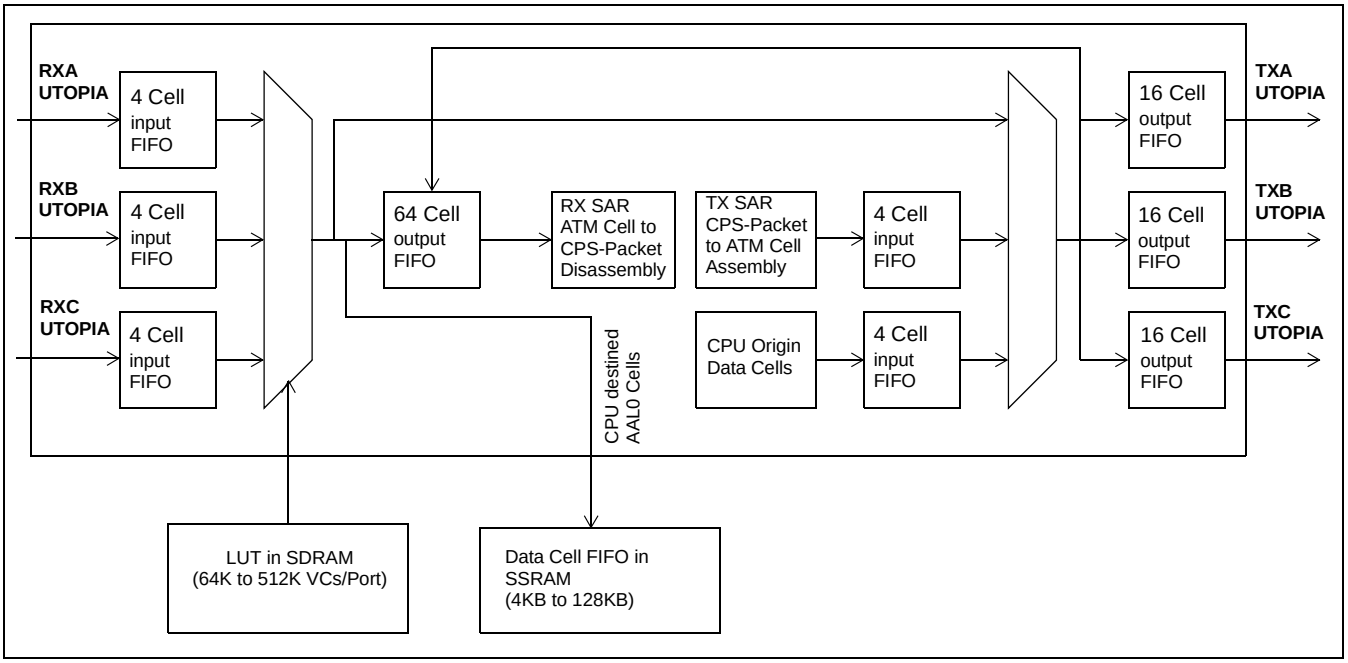



**Figure 34 - SAR and UTOPIA Block**

The receive interface of each UTOPIA port can be independently enabled or disabled. If disabled, the receive interface will stop accepting cells after the current cell has been received.

The transmit interface of each UTOPIA port can be configured to drive output pins (e.g. data bus, SOC and parity) only when this port has been selected. Those pins are tri-stated when the port is not selected. This allows the MT90502 to share a data bus, SOC, and parity lines with other devices (i.e. independent ENB signals and CLAV signals for each PHY device, controlled by a single ATM device).

### 2.6.2 UTOPIA Interfaces

Each of the three ports are divided into two portions: a receive portion and a transmit portion. The TX_SAR and the receive portions are each connected to a 4-cell FIFO.

The RX_SAR and the transmit portions are each connected to a 16-cell FIFO.

The ports are configurable with the following options:

- Port A's transmit portion can be ATM or PHY, with an 8-bit data bus.*
- Port A's receive portion can be ATM or PHY, with an 8-bit data bus.*
- Ports A and B can be combined to architect one UTOPIA Level 2 multi-PHY port, with an 8-bit data bus.
- Port B's transmit portion can be ATM or PHY, with an 8-bit data bus.*
- Port B's receive portion can be ATM or PHY, with an 8-bit data bus.*
- Port C's transmit portion can be ATM or PHY, with an 8-bit data bus.
- Port C's receive portion can be ATM or PHY, with an 8-bit data bus.
- Each receive interface can be independently enabled or disabled. If disabled, the receive interface will stop accepting cells after the current cell has been received.
- When the transmit portions of a port are in PHY mode, the SOC, data bus, and parity output pins will be tri-sected when the port is not selected. This allows the MT90502 to share a data bus, SOC, and parity lines with other devices (i.e. independent ENB signals and CLAV signals for each PHY device, controlled by a single ATM device).

*Not applicable when Port A and B are configured as level-2 multi-PHY port.

The RX_CLAV signal is asserted high any time a complete cell can be received. Thus as soon as the first byte of a cell is received, and there is no room for another cell in the input FIFO, the RX_CLAV signal will be asserted low. In the case of a Level-2 PHY, the rx_clav's will only be driven when the address has been placed on the bus during the previous cycle.

### 2.6.3 LED Operation

The UTOPIA module generates four LED signals in order to indicate the status of each of the possible conditions for the A and B ports. The status conditions are: idle, presence of traffic or PHY alarm. When a port is in an idle state, both its LEDs are illuminated. If RX traffic (other than null cells) is flowing, then the RX LED for that port will flash; If TX traffic (other than null cells) is flowing, then the TX LED for that port will flash. If a PHY alarm is detected, the TX LED is static on and the RX LED is static off. The polarity of the LED signals is active-low, i.e., a ‘0’ will turn on the LED.

The frequency of the LEDs is controlled in register 10Ch and the LED pins can also be configured to act as general purpose input outputs.

The PHY alarm is another testing feature. If a PHY device connected to the MT90502 indicates a trouble with the PHY (through pins rxa_alarm or rxb_alarm), the MT90502 acknowledges receipt of the information by setting the LEDs appropriately (see above). The alarm causes no change in the MT90502, other than from setting the LEDs.

### 2.6.4 Errors on Received Cells

If the MT90502 receives a short cell on any one of its three ports, the cell is discarded and the next cell is started when the second SOC signal is set.

If the SOC is not set after the 53rd byte of a received cell, subsequent bytes are ignored until a new SOC is received.

Data received on all three ports is examined for parity errors and an interrupt is raised if an error is found. Cells are not discarded if a parity error is detected. Register 60Ah indicates on which port the parity error is detected.

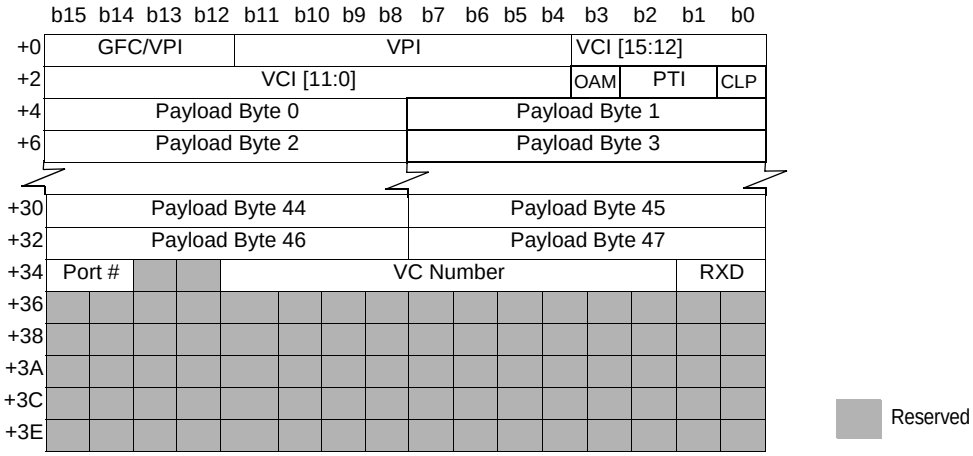
The ATM HEC is not examined on received cells.

### 2.6.5 Cell Routing

Cells are read in a cyclic manner from the 4-cell input FIFOs. Cells from the TX_SAR that contain a '0' in the MSB of the TXD field (see Figure 21, “TX Cell Format,” on page 43) are written into the output FIFOs designated by the three LSBs of the TXD field. The TXD field may also indicate that the cell is routed through one of the 3 look-up tables. Therefore, the cell follows the same path as if it came from one of the 3 UTOPIA input ports. Cells written into the RX_SAR FIFO can be directed to the SAR portion (cells to be formatted into TDM streams), to the data cell portion (to be examined by the CPU), or to both. This is indicated by the RXD field (see Figure 35, “RX Cell Format,” on page 64) in cells written into the RX_SAR output FIFO.

**RX Cell Format**

(of cells written in each of the internal input cell FIFOs in the UTOPIA interface)




RX SAR Output FIFO structure is 64-bytes in size, located in internal memory at addresses 3000h to 3FFEh.

**Port #:** “00” = comes from LUT A, “01” = comes from LUT B, “10” = comes from LUT C; others = reserved
**VC Number:** indicates the VC to which this cell is destined, if it is destined to the RX SAR.
**RXD:** RX Destination. “1x” = Destined to RX SAR, “x1” = destined to AAL0 FIFO.

**Figure 35 - RX Cell Format**

#### 2.6.5.1 Mask & Match Process

Cells received on the UTOPIA port are designated as “known” or “unknown” based on the result of a mask & match process (configured for each port). For a cell to be considered “known”, all VPI/VCI bits whose corresponding mask bit is '1' must have the value contained in the corresponding bit of the match register. In addition, the MT90502 can be configured to eliminate null cells (those with VPI = 0 and VCI = 0). For the purpose of null cell elimination, the NNI/UNI can be included on a per-port basis (Register 60Eh).

Unknown cells can be discarded or directed to one or more output FIFOs. All unknown non-OAM cells from a port are discarded or directed to the same location(s) and all unknown OAM cells from a port are discarded or directed to the same location(s). Unknown cells can be directed differently for each port on which they were

received. Unknown cells cannot be sent to the SAR portion of the RX_SAR. The routing of unknown cells is set in registers 6A4h and 6A6h.

Known cells are handled according to the LUT (Look-Up Table) entry for the cell's VPI/VCI.

A cell is deemed to be an OAM cell if its MSB of PTI field is set.

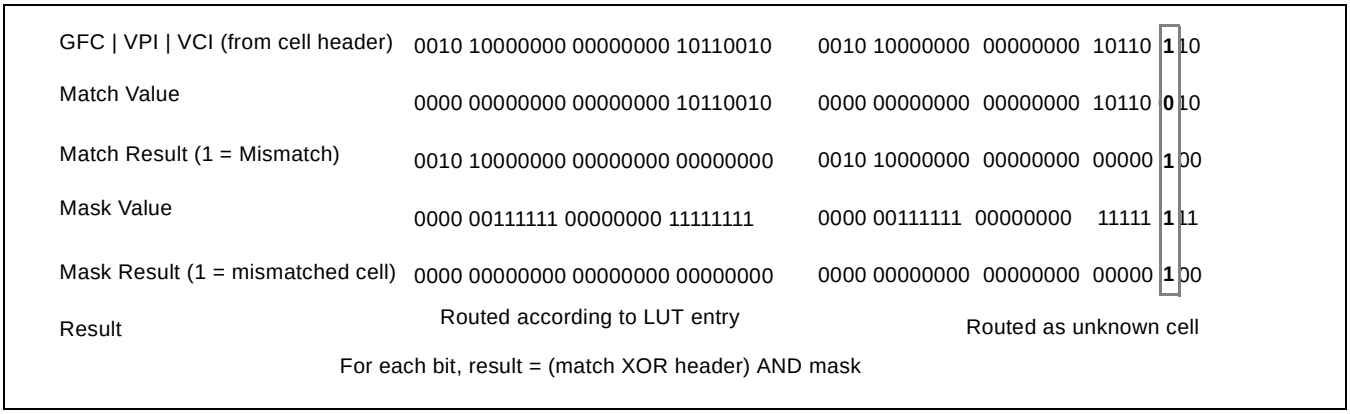


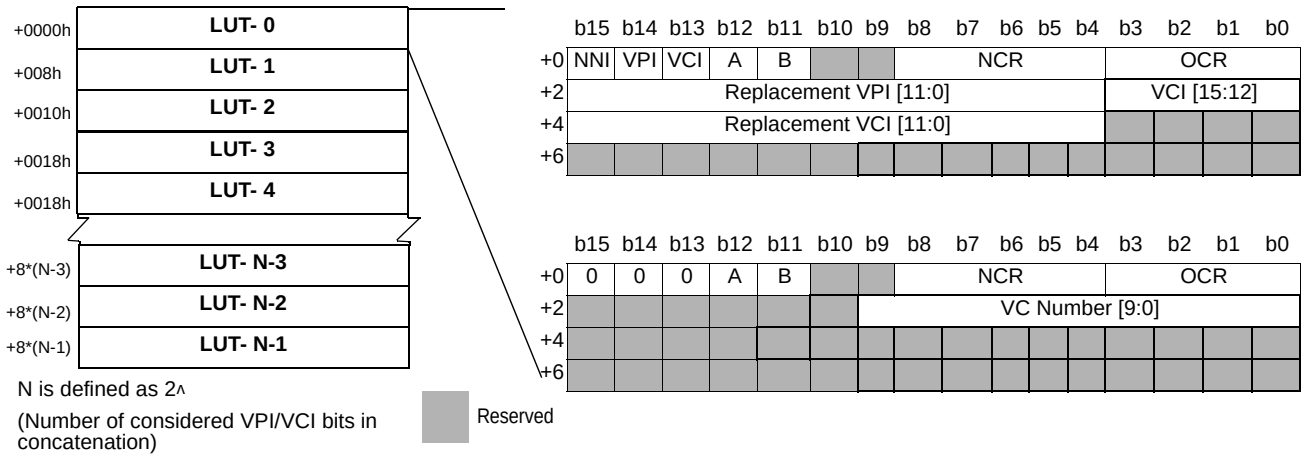

**Figure 36 - Mask & Match Example**

### 2.6.5.2 Look-Up Tables Entries

LUT entries direct cells with known VPI/VCIs to either be discarded or placed in one or more of five possible destinations: the four output FIFOs (UTOPIA A, UTOPIA B, UTOPIA C or RX SAR (internal)) and the data cell FIFO in external SSRAM. OAM cells can be directed independently of non-OAM cells with the same VPI/VCI. OAM cells cannot be directed to the SAR portion of the RX_SAR.

All look-up table entries in the three LUTs are the same size. Cells undergoing header translation have their NNI bits, the remaining VPI bits and/or the VCI bits replaced by the corresponding bits in the LUT entry and are then either discarded or sent to one or more of the possible destinations. Note: the 4 MSBs of the header (VPI bits 11:8 in NNI mode, or the GFC field in UNI mode) can be translated separately from the remaining portion of the VPI. The remaining two portions can also be translated separately, 8 bits of VPI and the 16 bits of VCI. The 3 header translation enable bits, NNI, VPI, and VCI denote the translation portion of the header. VCs that undergo header translation cannot be directed to the SAR portion of the RX_SAR as no RX_SAR structure pointer is available in the LUT entry.

The look-up engine also contains bits indicating if the received cell is a timing reference cell. A timing reference cell indicates a cell whose arrival frequency, over a long period of time, is constant, and therefore can be used as a reference for the local TDM clock. If a timing reference cell is received, the UTOPIA module will send a pulse to the clock recovery module, indicating that such a cell has been received. Clock recovery information can be gathered from up to two VCs by setting bit A in one LUT entry and bit B in the same or another LUT entry. A maximum of one VC can have bit A set and a maximum of one VC can have bit B set.




LUT A,B, and C structures are 8-bytes in size located in SDRAM at varying base addresses depending on the number of channels used.

Fields in Plain are written by the CPU/Software.

**NCR:** Normal Cell routing. “00000” = discard; “xxxx1” = Send to TXA port; “xxx1x” = Send to TXB port; “xx1xx” = Send to TXC port; “x1xxx” = Send to data cell FIFO (in external memory); “1xxxx” = Send to RX SAR.

**OCR:** OAM Cell Routing. “0000” = discard; “xxx1” = Send to TXA port; “xx1x” = Send to TXB port; “xx1xx” = Send to TXC port; “1xxx” = Send to data cell FIFO (in external memory).

**A & B:** Clock recovery reference indicators. When “1”, a pulse will be sent to the clock recovery module indicating that a reference packet has been received. These two bits correspond to two independent and redundant clock recovery VCs.

**VC Number:** Points to the RX Disassembly Structure to which this cell corresponds.

**NNI:** Replace the cell VPI [11:8] with the Replacement VPI [11:8] when this bit is “1”.

**VPI:** Replace the cell VPI [7:0] with the Replacement VPI [7:0] when this bit is “1”.

**VCI:** Replace the cell VCI [15:0] with the Replacement VCI [15:0] when this bit is “1”.

**Figure 37 - SDRAM Mapping - Look-Up Tables Structure**

### 2.6.5.3 LUT Addressing

A LUT base address and size exists for each of the three ports (registers 620h, 640h, 660h). The number of entries in the LUT, given by its size, ranges from 17 bits to 20 bits (128K to 1024K entries). A 17- to 20-bit identifier for each entry is created by concatenating the LSBs from the VPI field with LSBs from the VCI field of a cell header. The number of VCI LSBs to be used as the LSBs in the identifier is programmed in registers 622h, 642h and 662h for Port A, B and C, respectively. The remaining MSBs of the identifier are taken from the LSBs of the VPI field. This concatenated value is then multiplied by 8 (insert three LSB zeros). The result, when added to the LUT base address, will point to the base address of the LUT structure for this cell. The LUT, shown in Figure 37 will determine the routing and/or header translation of the cell.

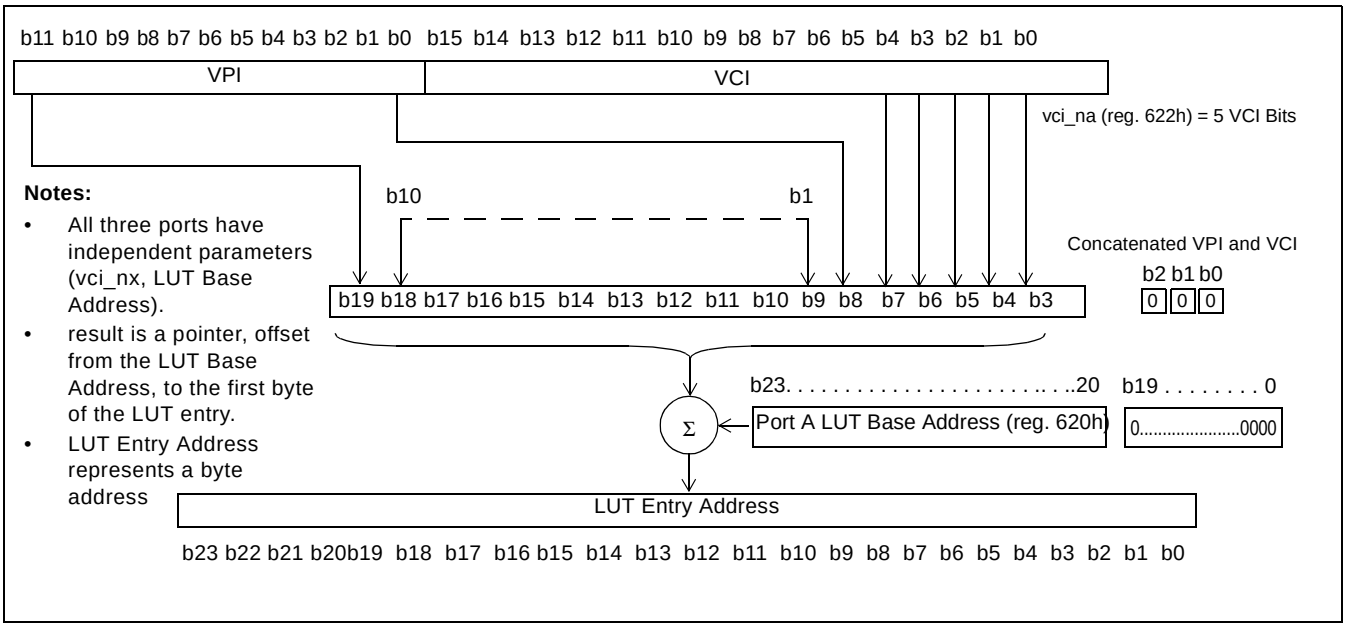



**Figure 38 - VPI/VCI Concatenation and LUT Entry Address Example**

### 2.6.6 UTOPIA Clocks

Each of the three ports must have a clock to operate the receive interface and a clock to operate the transmit interface. Two or more clocks may have the same source. These clocks can either be input to the MT90502 from an external source or output from the MT90502 from one of three internal UTOPIA clocks. However, the receive clock and transmit clock for a port can either be set as an input or an output.

The source of each of the three internal UTOPIA clocks can be one of seven clocks: mem_clk, or any of the six UTOPIA clocks (rxa_clk, rxb_clk, rxc_clk, txa_clk, txb_clk, and txc_clk). The selected clock is divided by n and can be inverted (register 220h).

The maximum speed of UTOPIA clock is 32MHz.

Other parts of the UTOPIA module, including the look-up engine, the TX_SAR portion, and the RX_SAR portion operate off of mem_clk.

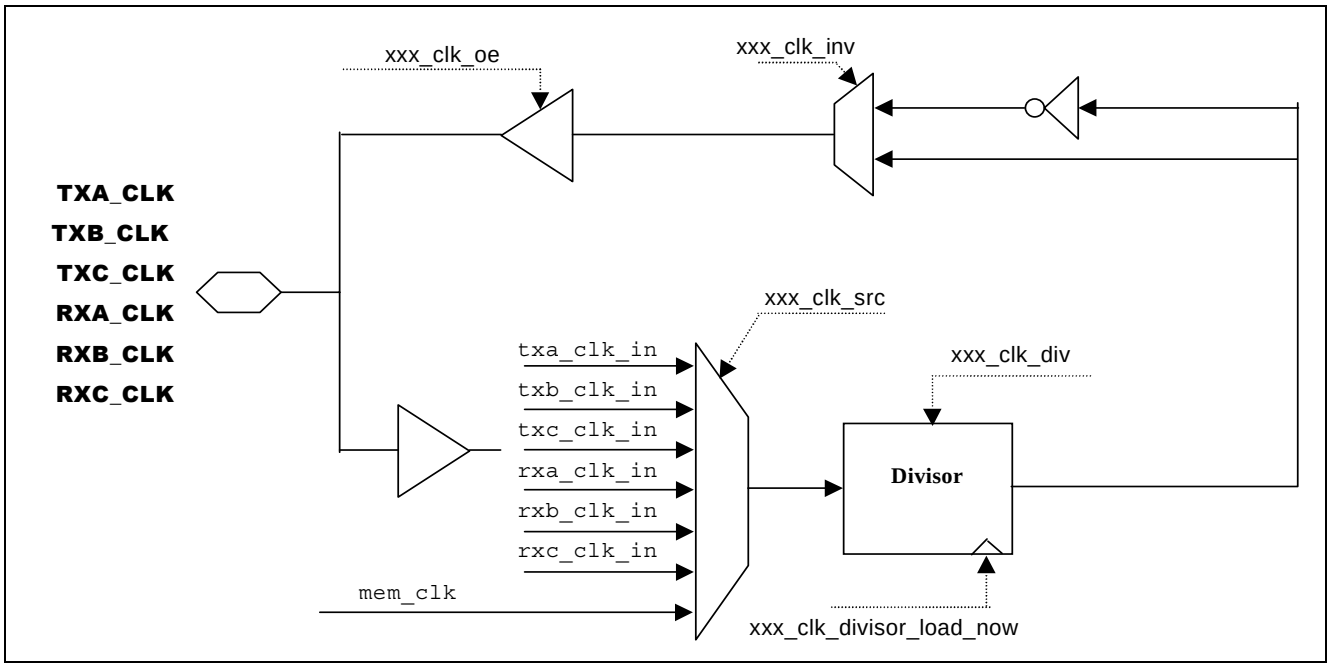



**Figure 39 - UTOPIA Clocks Selection**

| Label | Register | Bit Position | Label | Register | Bit Position |
|---|---|---|---|---|---|
| txa_clk_div | 220h | 5:0 | rxa_clk_div | 228h | 5:0 |
| txa_clk_divisor_load_now | 220h | 6 | rxa_clk_divisor_load_now | 228h | 6 |
| txa_clk_inv | 220h | 7 | rxa_clk_inv | 228h | 7 |
| txa_clk_src | 220h | 10:8 | rxa_clk_src | 228h | 10:8 |
| txa_clk_oe | 220h | 11 | rxa_clk_oe | 228h | 11 |
| txb_clk_div | 222h | 5:0 | rxb_clk_div | 22Ah | 5:0 |
| txb_clk_divisor_load_now | 222h | 6 | rxb_clk_divisor_load_now | 22Ah | 6 |
| txb_clk_inv | 222h | 7 | rxb_clk_inv | 22Ah | 7 |
| txb_clk_src | 222h | 10:8 | rxb_clk_src | 22Ah | 10:8 |
| txb_clk_oe | 222h | 11 | rxb_clk_oe | 22Ah | 11 |
| txc_clk_div | 224h | 5:0 | rxc_clk_div | 22Ch | 5:0 |
| txc_clk_divisor_load_now | 224h | 6 | rxc_clk_divisor_laod_now | 22Ch | 6 |
| txc_clk_inv | 224h | 7 | rxc_clk_inv | 22Ch | 7 |
| txc_clk_src | 224h | 10:8 | rxc_clk_src | 22Ch | 10:8 |
| txc_clk_oe | 224h | 11 | rxc_clk_oe | 22Ch | 11 |

**Table 25 - UTOPIA Clocks Selection Registers**

### 2.6.7 External Interface Signals

Due to the different possible configurations of the UTOPIA ports, the functions of some pins change, depending on the configuration. The function of the CLAV (cell available) and ENB (enable data transfer) pins alternate when the port is in ATM mode or PHY mode (see Figure 40, “External UTOPIA Interface,” on page 69). Please note that the I/O direction of the pins remain fixed.

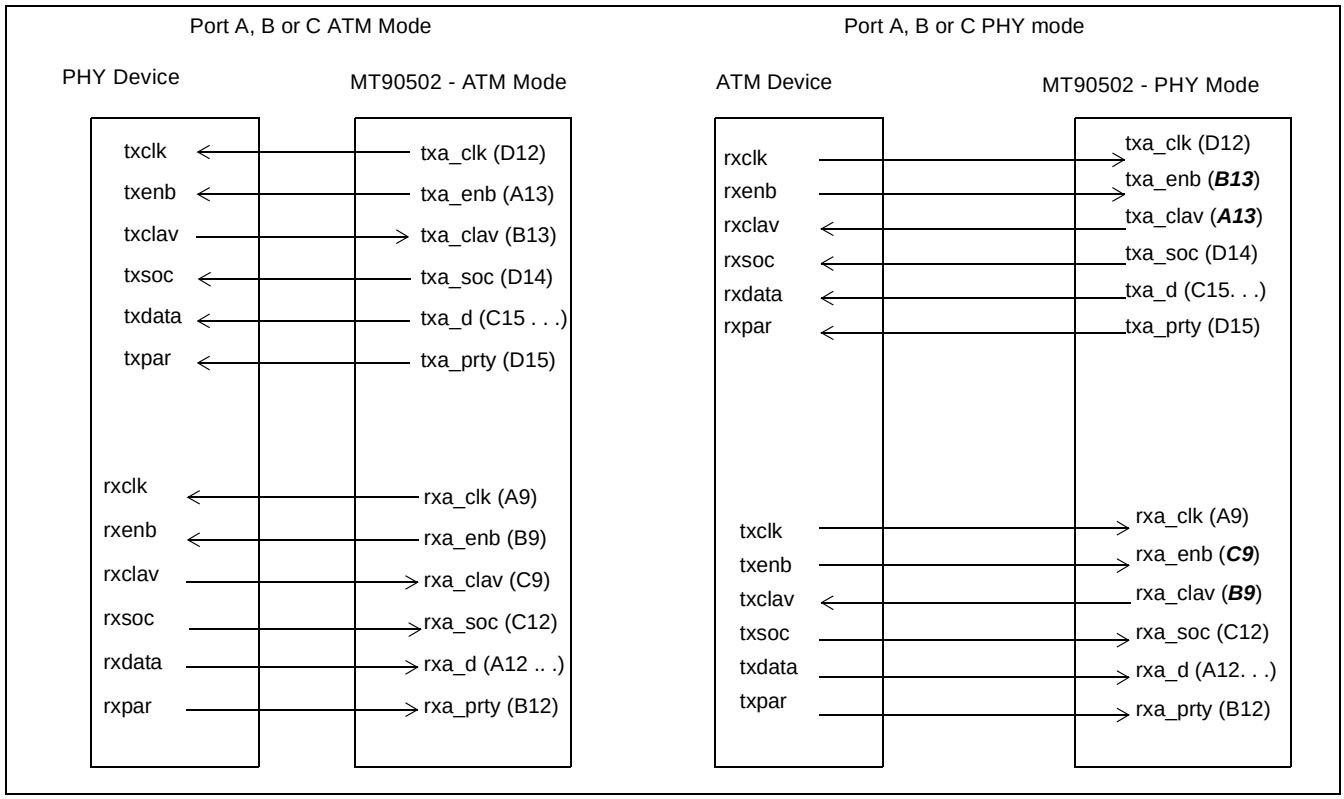



**Figure 40 - External UTOPIA Interface**

### 2.6.8 UTOPIA Flow Control

The UTOPIA module contains the ability to prevent cells in the 4-cell input FIFOs (RXA, RXB, RXC, TX Data Cell FIFO and TX_SAR) from being processed by the UTOPIA module in case the output FIFOs (TXA, TXB, TXC and RX_SAR) exceed programmable levels.

For each pair of input and output FIFO, a threshold can be set independently from 0 to 15 (for TXA, TXB and TXC output FIFOs) or 0 to 63 (for RX_SAR output FIFO). Once the cell level in output FIFO exceeds that threshold, cells from that input FIFO are blocked until the output FIFO empties itself below the threshold. There are cell arrival counters and cell departure counters for each port.

## 2.7 H.100/H.110 Interface

### 2.7.1 Overview

The H.100/H.110 interface is compatible with the ECTF H.100/H.110 hardware compatibility specification, Computer Telephony BUS (CT-BUS), and its implementation on the PCI computing platform. The TDM interface of the MT90502 can be used to interface as bus master or bus slave with the H.100/H.110 bus. The MT90502 supports up to 32 TDM Streams running at 8.192MHz (up to 4096 time slots). Also, as required in the specification, 16 TDM streams can be configured to run at lower frequencies of 4.096MHz or 2.024MHz. The MT90502 can perform loopback on 128 channels.

### 2.7.2 Bus Signaling

There are four classes of signals on the H.100/H.110 CT-Bus: Core, Compatibility, Optional, and Reserved signals. Core, Compatibility, and Reserved signals are required for all H.100/H.110 compatible devices. These signals are described in Table 26, “CT-Bus Signalling Function,” on page 70.

### 2.7.3 H.100/H.110 Slave

| Class | Signal | Function |
|---|---|---|
| Core | CT_FRAME_A | Frame Synchronisation, driven by the “A” clock master. |
| | CT_C8_A | Bit clock driven by the “A” clock master. |
| | CT_FRAME_B | Configurable frame synchronisation, driven by the “B” clock master. |
| | CT_C8_B | Configurable bit clock - driven by “B” clock master. |
| | CT_D[31:0] | Serial Data lines that collectively carry 32 signals and are referred to as the CT_D bus. Each signals contain 128 time slots per frame at a clock frequency of 8.192MHz. |
| | CT_NETREF | Secondary network timing reference, providing backup network synchronisation to the CT Bus. |
| Compatibility | FR_COMP | Compatibility frame pulse, driven by current clock master, that serves as the frame synchronization signal for the SCBus (Fsync*) and MVIP (F0) signals. |
| | SCLK | SCBus system clock, driven by current clock master. The signal is selectable (2, 4, or 8MHz) and is used to identify the data bits positions on the SCBus. |
| | SCLKx2 | SCBus system clock times two, driven by clock master. |
| | C2 | MVIP-90 bit clock, driven by current clock master. The clock frequency is 2MHz, nominally symmetrical. |
| | C4 | MVIP-90 bit clock times two, driven current by clock master. The clock frequency is twice C2, and transitions of C2 are synchronous with the falling edge of C4. |
| | C16+, C16- | H-MVIP 16 MHz Clock, driven by current clock master. This differential signal is used to read and write bits on the serial data lines by H-MVIP boards. |
| Optional | CT_MC | Message Channel for inter-device communication. This signal is terminated on each CT Bus interface in the system which has message bus capability. |
| | CT_+5Vdc | Provides power to active transition devices (cable adapters). |
| Reserved | | Signals reserved for future use. |

**Table 26 - CT-Bus Signalling Function**

The H.100/H110 interface can sample the incoming data on the ct_d[31:0] at different sampling points: 2/4, 3/4 or 4/4 sampling. The desired sampling point can be set by setting the timing configuration register (732h). When transmitting data, the H.100/H.110 can tri-state its pins between 20ns and 0ns before the rising edge of

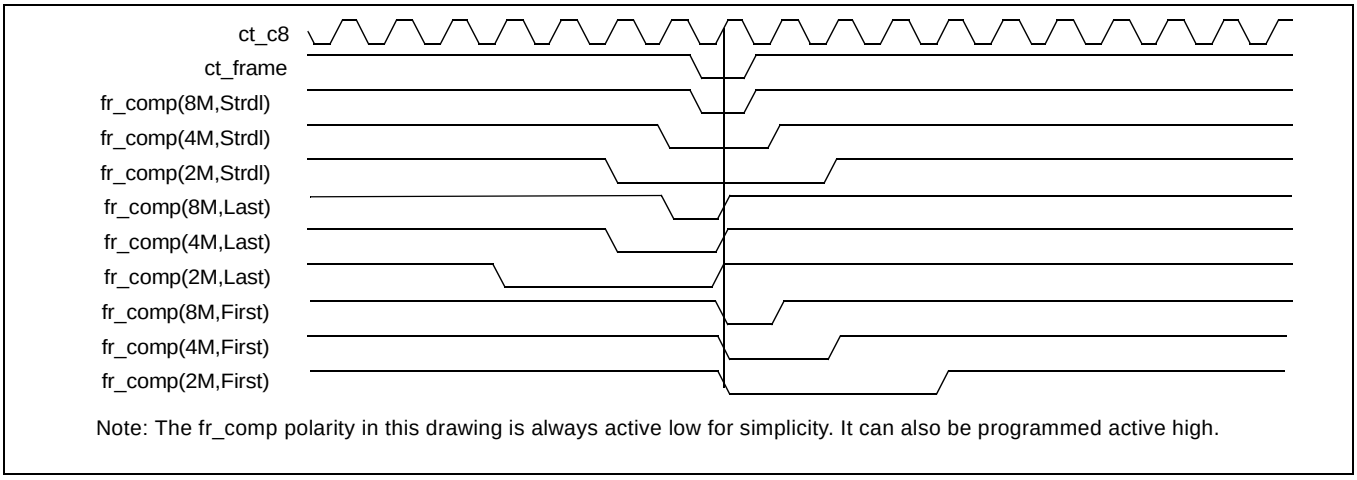
the clock or it can tri-state synchronously on the rising edge of the clock. Both of these options allow flexibility in compatibility with other devices that are not fully H.100/H.110 compliant.

### 2.7.4 Operating as a Slave

Two clocks, A and B, can be configured to set one clock as primary bus master and the other one as backup master. In the event of primary clock failure, the interface can switch over to backup. The MT90502, when operating as a slave, has the choice of clocking on A, clocking on B, clocking on A with B as backup, or clocking on B with A as backup. When set to perform automatic switch-over, the interface monitors the active bus master to determine if its ct_c8 clock edges are within ±35 ns of their expected edge times (122 ns apart) and the ct_frame signal occurs upon the 1024 ct_c8 clock cycle. If ct_c8 or ct_frame errors are reported on the bus master signals, the ct_c8 or ct_frame is considered invalid and the slave will switch to the backup master (if backup has been configured). The MT90502 will always monitor these signals and will report errors on either of the two bus masters in status0 register (702h).

### 2.7.5 Operating as a Master

When configured as a bus master, the MT90502 can be a bus master on A, B, backup on A or backup on B. The difference between a bus master and a bus backup is that bus master drives all compatibility clocks, and the bus backup does not. When configured as a bus backup, the MT90502 uses the same error signals or error flags to determine if the current bus master is valid or if MT90502 reverts to be the master. Note that the bus mastership can be overridden in registers mastership_hidden0 (774h), mastership_hidden1 776h) and mastership_hidden2 (778h) by ensuring that the MT90502 cannot drive the H.100/H.110 clock and frame signals. If the MT90502 is a backup on the bus and the primary master fails, it will stop synchronising to the master and track the local reference.




Note: The fr_comp polarity in this drawing is always active low for simplicity. It can also be programmed active high.

**Figure 41 - TDM Bus Timing - Fr_Comp Generation**

Once configured as bus master, MT90502 provides compatibility clocks generated by H.100/H.110 module. Figure 42, “TDM Bus Timing - sclkx2 Generation,” on page 72 shows the MT90502’s compatibility signals generated on H.100/H.110 interface.

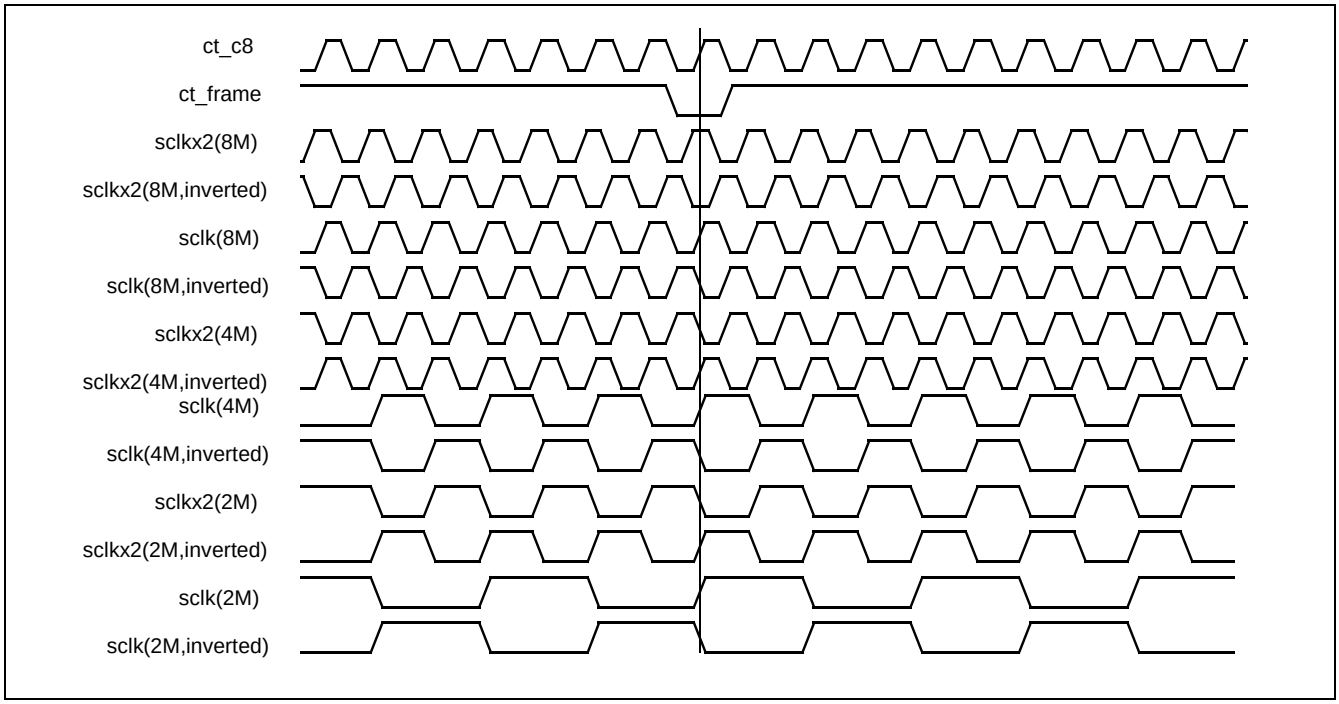



**Figure 42 - TDM Bus Timing - sclkx2 Generation**

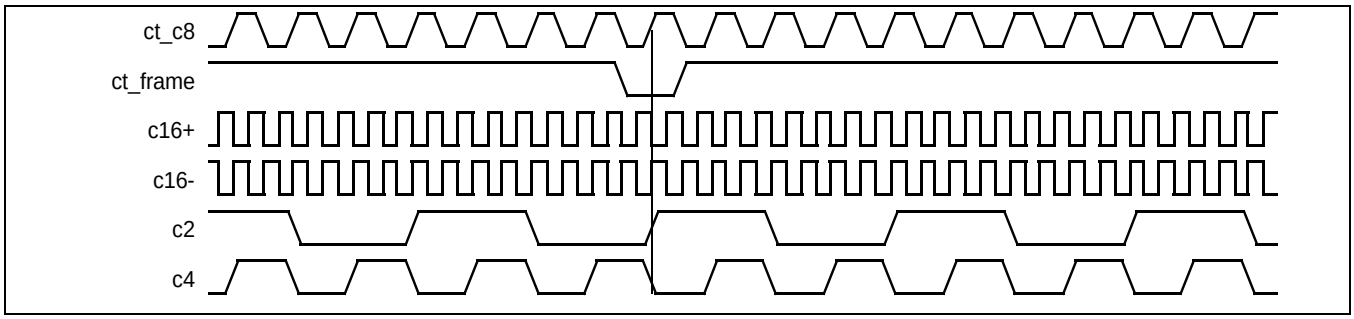



**Figure 43 - TDM Bus Timing - Compatibility Clock Generation (other than sclk, sclkx2)**

The sclk and sclkx2 signals can be programmed in mastership0 register (720h) to have identical or opposite polarities as defined in the H.100/H.110 specifications. In addition, the frequency of sclk can be programmed to be 8.192MHz, 4.096MHz or 2.048MHz.

The lower 16 streams on the H.100/H110 bus are grouped into 4 and are capable of operating at the following frequencies: 8.192MHz, 4.096MHz, or 2.048MHz. The MT90502 can be programmed such that all 32 streams are arranged in groups of 4 streams - ct_c[3:0], ct_c[7:4], ct_c[11:8], ct_c[15:12], ct_c[19:16], ct_c23:20], ct_c[27:24] and ct_c[31:28]. Each group can be assigned a desired frequency. These features are programmed in clock_rates register (730h).

In addition, the MT90502, instead of supporting the full bandwidth of H.100/H.110, can be configured to only interface with 4, 8 or 16 streams on the bus. This allows the MT90502 to operate at lower frequencies. The lowest streams are used in these cases.

### 2.7.6 H.100/H.110 Clock Selection Guide

Fast_clk is used internally by the MT90502 as its main operating clock. The internal logic used to generate fast_clk is show in Figure 44 on page 73. Typically mem_clk would be selected using pll_source.

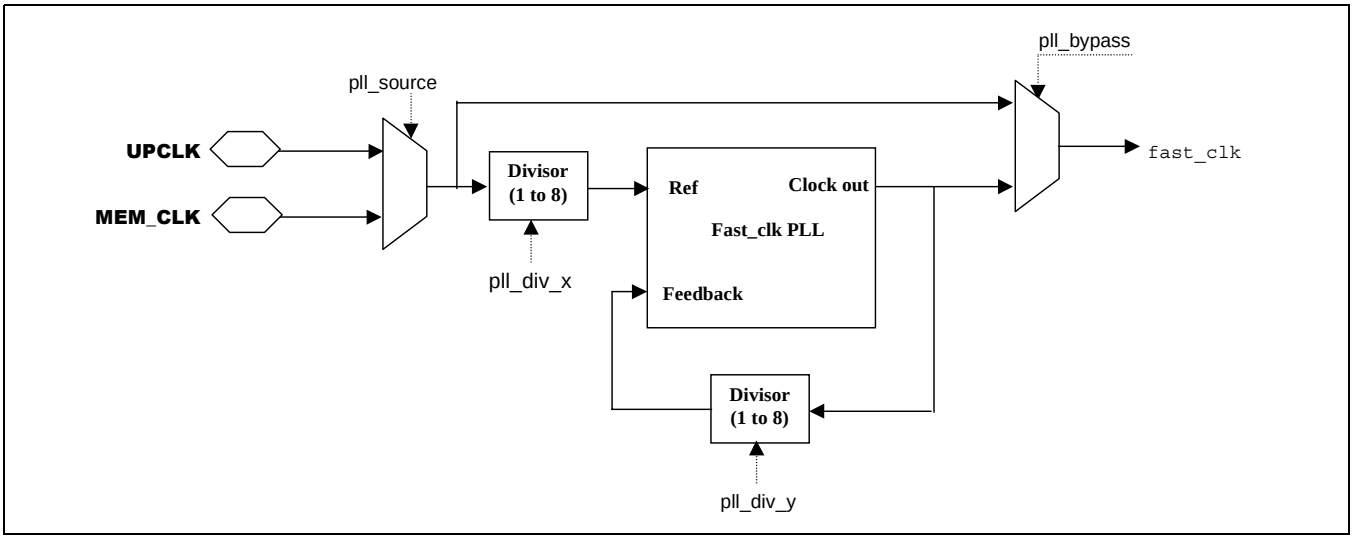



**Figure 44 - Fast Clock Generation**

The pll_div_x and pll_div_y should be programmed according to Figure 44, “Fast Clock Generation,” on page 73. The fast_clk frequency is given by the formula

$$\text{fast_clk} = \text{input_frequency} \times \frac{\text{pll_div_y}}{\text{pll_div_x}}.$$

| upclk/mem_clk (MHz) | pll_div_x | pll_div_y |
|---|---|---|
| >75 | 1 | 2 |
| >66 | 2 | 5 |
| >50 | 1 | 3 |
| >40 | 1 | 4 |
| >30 | 1 | 5 |
| >25 | 1 | 6 |

**Table 27 - Fast Clock PLL Divisor Values**

| Label | Register | Bit Position |
|---|---|---|
| pll_div_x | 16Ch | 3:1 |
| pll_div_y | 16Ch | 6:4 |
| pll_bypass | 16Ch | 7 |
| pll_source | 16Ch | 8 |

**Table 28 - Fast Clock Registers**

The mastership mode and slaveship mode of the MT90502’s TDM interface can be programmed using register 722h. Mastership mode determines on which bus(es), A or B, the MT90502 will act as a master or backup. When configured as a master on a bus, the MT90502 drives the frame pulse and clock on that bus as well as the compatibility signals (if enabled). When configured as a backup on a bus, the MT90502 drives the frame

pulse and clock on that bus, but does not drive the compatibility signals. The slaveship mode configures the MT90502 to synchronize its internal timing to the ct_frame pulse and ct_c8 clock on either the A or the B bus. The slaveship mode is completely independent of the mastership mode. An example configuration would have the mastership mode set for the MT90502 to be a master on A and setting the slaveship mode for the MT90502 to track on A.

Register 774h can be used to override the operation of the MT90502 based on the selections made in register 722h. The automatic_master_override bit will enable the rest of the values in this register. The mux0_select_override bit will stop the automatic fallback of the slaveship mode (from A to B or B to A). The mux1_select_override bit will select between using ct_c8_a/ct_c8_b or H.110 PLL output clock as the feedback signal to the H.110 PLL. The mux2_select_override bit will select between ct_c8_a/ct_c8_b or pll_clk as the reference source for the H.110 PLL. The mux3_select_override bit will select between using either an external pll_clk or the H.110 PLL output as the clock used to generate the ct_c8_x and ct_frame_x and clock signals.

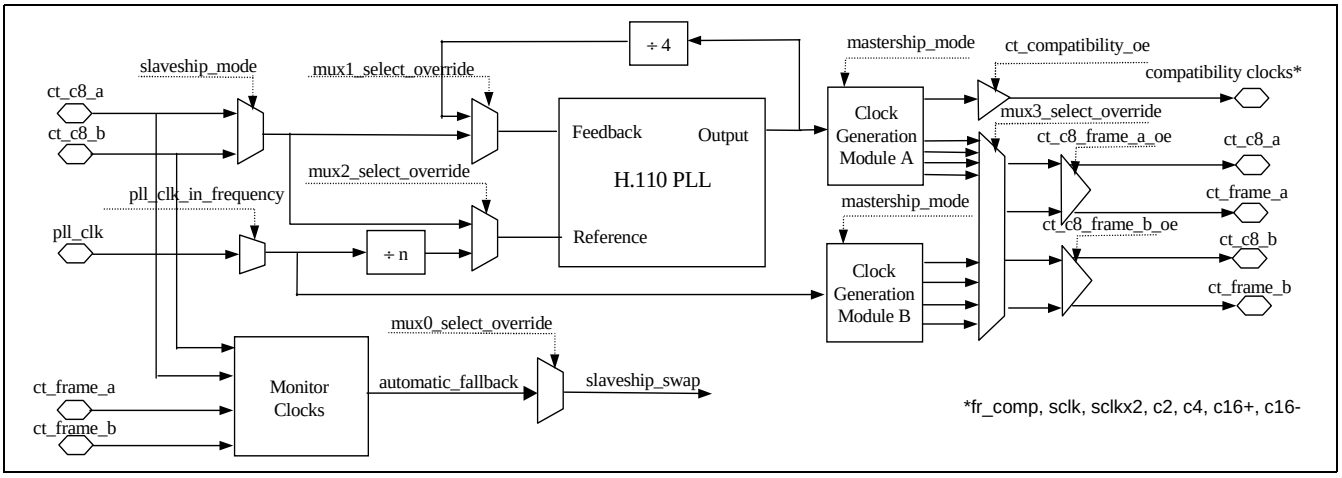



**Figure 45 - H.100/H.110 PLL Clock Selection**

| Label | Register | Bit Position |
|---|---|---|
| pll_clk_in_frequency | 700h | 1:0 |
| ct_c8_frame_a_oe | 720h | 0 |
| ct_c8_frame_b_oe | 720h | 1 |
| ct_compatability_oe | 720h | 2 |
| mastership_mode | 722h | 1:0 |
| slaveship_mode | 722h | 3:2 |
| automatic_master_override | 774h | 0 |
| mux0_select_override | 774h | 1 |
| mux1_select_override | 774h | 2 |
| mux2_select_override | 774h | 3 |
| mux3_select_override | 774h | 5:4 |

**Table 29 - H.100/H110 PLL Clock Selection Registers**

## 2.8 Clock Recovery

### 2.8.1 Overview

The purpose of the clock recovery module is to synchronize the TDM clock domain of the MT90502 with other devices on the network through information transmitted across the ATM link. Clock recovery is necessary only when the MT90502 is operating as the TDM clock master.

The clock recovery system is composed of several sub-components:

#### 2.8.1.1 Adaptive Clock Recovery Modules

These modules generate data on a regular basis which is written to external memory for the CPU to read and is used by the clock recovery algorithm. Counts of mem_clk and adapx_ref clock, as well as the current UUI and LI values, associated with the current cell, are placed in external memory.

##### 2.8.1.1.1 adapx_ref clock generation

These modules divide mem_clk to a desired frequency with a 16-bit integer and 16-bit fraction. These 32 bits are normally determined by the clock recovery algorithm.

#### 2.8.1.2 Multiplexers

There are eight general purpose I/Os as well as two ct_netref pins. Each has the ability to multiplex one of 23 signals as outputs.

### 2.8.2 Adaptive Clock Recovery Modules

The MT90502 has two clock recovery modules, A and B. Each module receives pulses obtained either from the UTOPIA module (timing reference VCs) or from the Rx SAR (timing reference CPS-Packets). The signals clkrecov_pulse_a and clkrecov_pulse_b come from one of these modules. The modules which generate the A and B clkrecov_pulses are configured by the A and B bits. These are configured in the UTOPIA section for timing reference VCs or in the Rx SAR section for timing reference CPS-Packets. The VC/CPS-Packet designated A is the timing reference for the A clock recovery module (clkrecov_pulse_a). Similarly, for the B clock recovery module, the clkrecov_pulse_b is generated by the VC/CPS-Packet designated by its B bit.

Running in parallel, are counters of the mem_clk and adapx_ref signals. A clock recovery event structure is written to external memory with the arrival of every X pulses of clkrecov_pulse_x. The structure (see Figure 46 on page 76) is composed of the 32-bit “mem_clk” counter, the 32-bit “ref” counter, a 16-bit “mem_clk_cycles_since_last_ref_increment” counter (fraction of the ref), as well as the LI and UUI of the received CPS-Packet. These event structures are written to a buffer in external memory, and the clock recovery module will generate an interrupt when the buffer is more than half full (if enabled in bit 10 of register 210h). Clock recovery events arrive at a fixed rate, therefore the size of the buffer chosen (820h, 828h Table 31, “Buffer Sizes,” on page 81) will completely determine the rate at which it needs to be serviced. To decrease the number of points written to memory, program the keep_one_pulse_out_of_x register (710h & 718h see Table 30, “Clock Recovery Registers,” on page 79) to a value greater than 1.

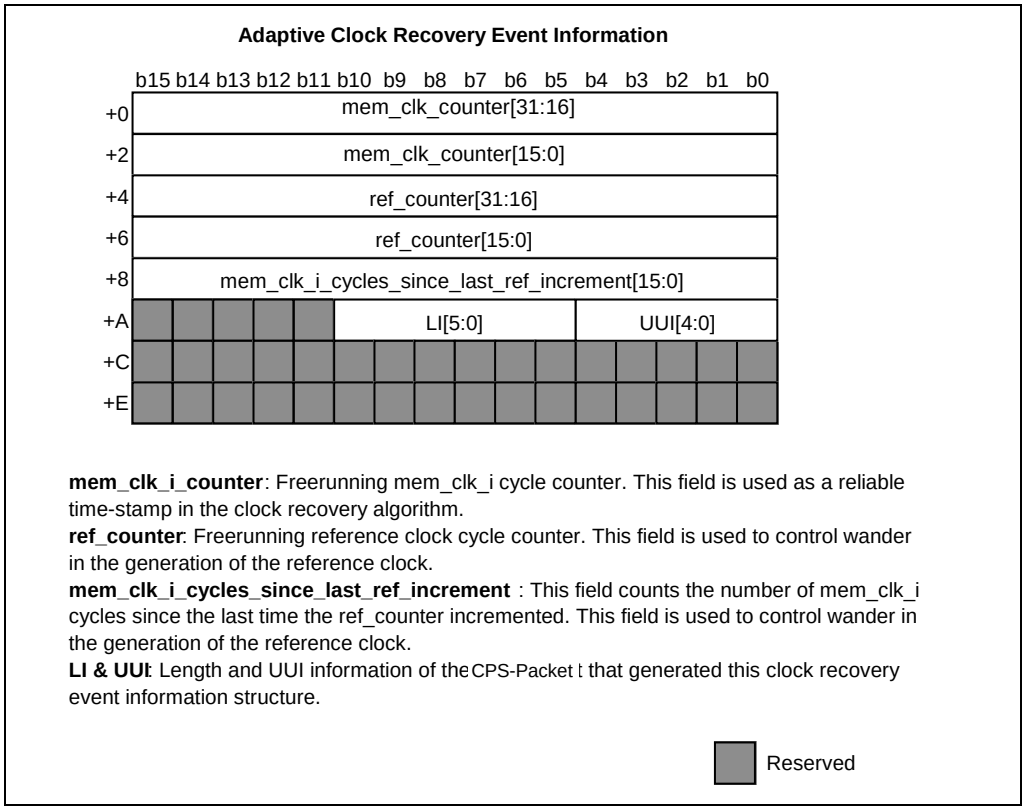



**Figure 46 - Adaptive Clock Recovery Event Information**

The base addresses and structures sizes for adaptive clock recovery event structures for module A and module B are programmable in registers 820h ‘pointa_manage’ and 828h ‘pointb_manage’ respectively. The address of the structures to be read can be determined using the read and write pointers located in registers 822h ‘pointa_read’, 824h ‘pointa_write’, 82Ah ‘pointb_read’ and 82Ch ‘pointb_write’.

Similar to the event/error FIFO and the data cell FIFO, the read pointers must be updated by the CPU/software and the write pointers are updated by the hardware.

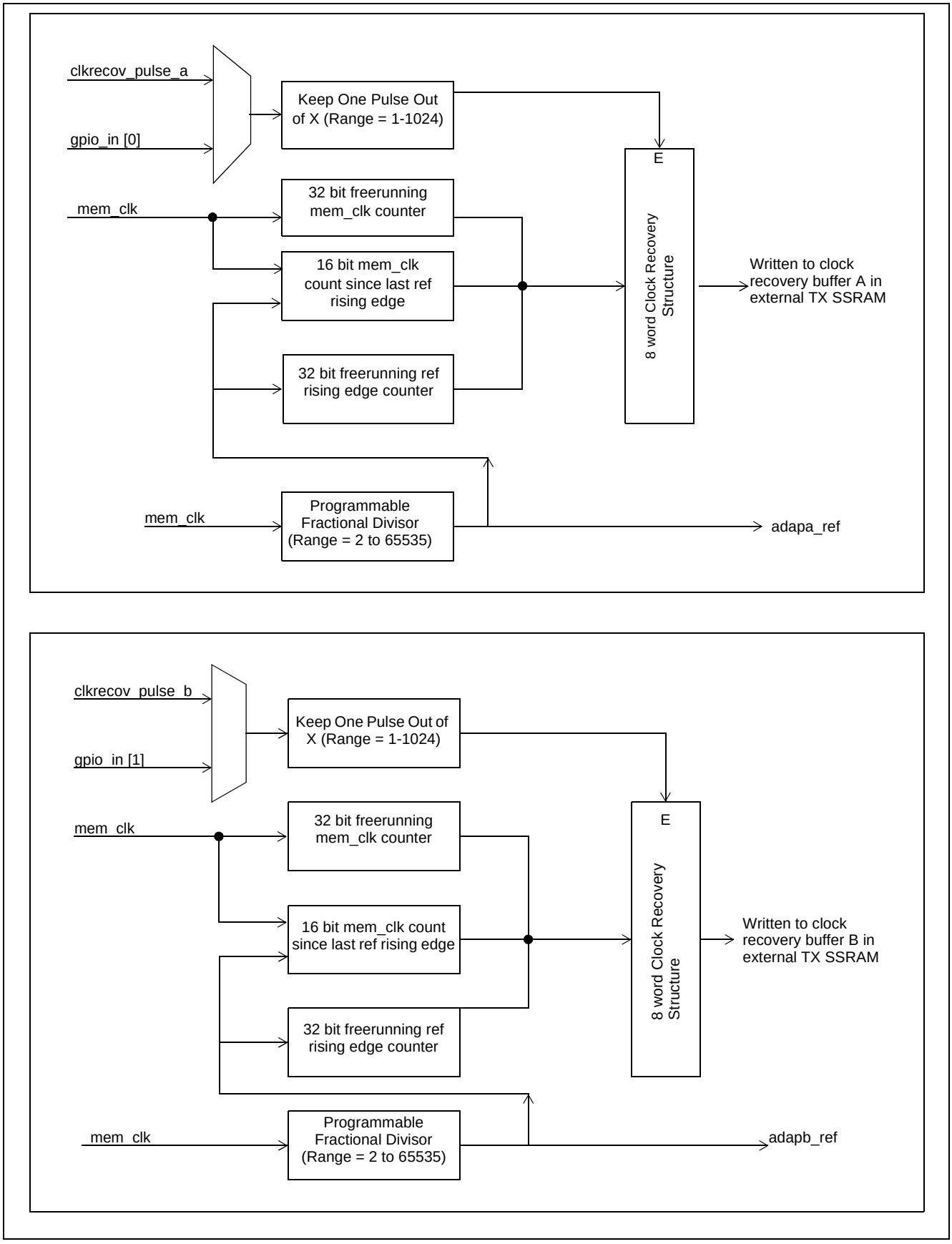



**Figure 47 - Adaptive Clock Recovery Modules**

#### 2.8.2.1 adapx_ref Clock Generation

The second portion of the clock recovery circuit is concerned with generating the adapx_ref signal. When the CPU reads the information placed in the clock recovery event buffer, the clock recovery algorithm can subsequently calculate any correction required to the rate of the adapx_ref clock. A typical configuration would see the adapx_ref signal operating at 8kHz. This is outputted to a PLL which multiplies the frequency up to 16.384MHz. Finally, the 16MHz clock is used as the driving clock for the TDM section, pll_clk (pin AB2).

The adapx_ref clocks are generated by dividing mem_clk by a 16-bit integer (712h & 71Ah) and 16-bit fraction (714h & 71Ch). This allows a highly precise division (with mem_clk running at 50 MHz and the target clock speed 8 kHz, it gives a precision of 2.4 ppb).

$$f_{\text{adapx_ref}} = \frac{f_{\text{mem_clk}}}{integer + \frac{fraction}{65536}}$$

The fractional divider will introduce jitter into the adapx_ref signal of a maximum of one mem_clk cycle. If it is desired that no jitter be added by the adapx_ref module, set fraction to 0 (this will reduce precision to 160 ppm from 2.4ppb). The integer and fraction are programmed by the clock recovery algorithm. The adapx_ref signal can be driven onto any of the eight GPIOs on the MT90502, or to one of the ct_netrefs (see Figure 49). It can be externally routed to a PLL used to multiply it up from 8 kHz to 16.384 MHz and then rerouted into the MT90502 on the pll_clk pin.

### 2.8.3 Multiplexers

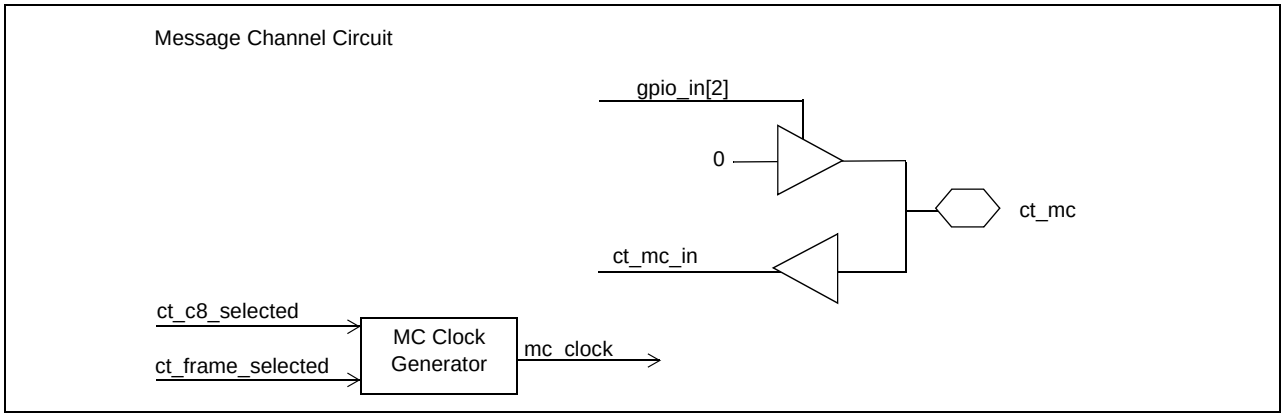



**Figure 48 - Message Channel**

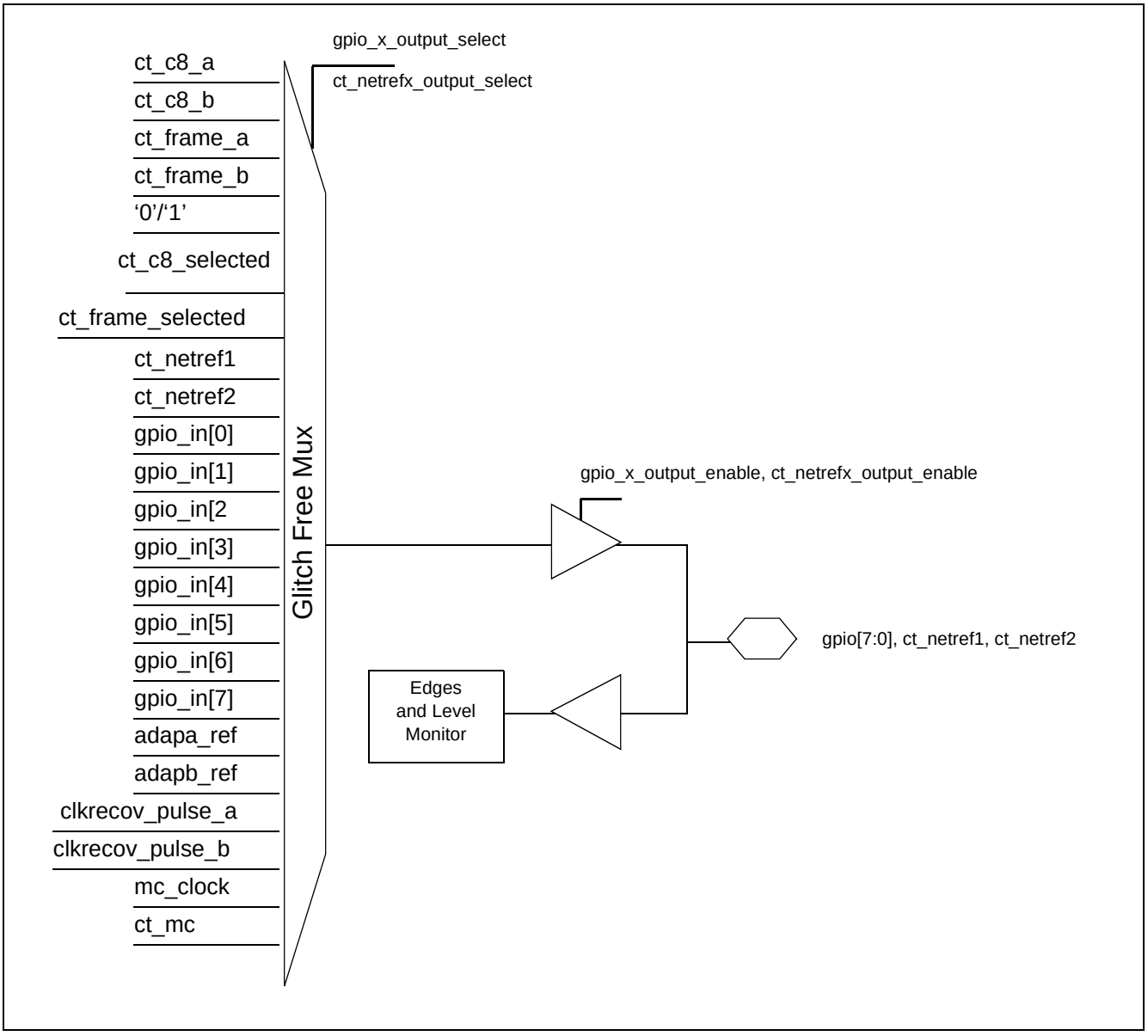



**Figure 49 - GPIO Functionality**

There are eight GPIO multiplexers in the clock recovery section of the MT90502. In addition, there are two multiplexers for the two ct_netref signals of the H.110 bus. See registers 740h - 748h (Table 30, "Clock Recovery Registers," on page 79). There are 23 possible inputs for each multiplexer (see Table 32, "GPIO mux, Output Selection," on page 82).

| Address | Bits | Register | Bits | Description |
|---|---|---|---|---|
| 210h | [10] | interrupt_flags | clkrecov_alarm_interrupt_active | Indicates clock recovery buffer is half full. |
| 218h | [10] | interrupt_enable1 | clkrecov_alarm_interrupt1_enable | Activates interrupt1 when clkrecov_alarm set. |
| 21Ah | [10] | interrupt_enable2 | clkrecov_alarm_interrupt2_enable | Activates interrupt2 when clkrecov_alarm set. |
| 710h | [9:0] | adapa0 | keep_one_pulse_out_of_x | Number of points generated to send to external memory. |

**Table 30 - Clock Recovery Registers**

| Address | Bits | Register | Bits | Description |
|---|---|---|---|---|
| | [10] | | clk_divisor_reset | If ‘0’, reset clock divisor. |
| | [12:11] | | source | ‘00’ - reference VC, ‘01’ - gpio[0] all edges, ‘10’ gpio[0] rising edges, ‘11’ gpio[0] falling edges |
| | [13] | | clk_divisor_load_now | If set to ‘1’, loads adaptive module A’s divisor. |
| 712h | [15:0] | adapa1 | div_integer | Adaptive module A - integer divisor |
| 714h | [15:0] | adapa2 | div_fraction | Adaptive module A - fractional component of divisor |
| 718h | [9:0] | adapb0 | keep_one_pulse_out_of_x | Number of points generated to send to external memory. |
| | [10] | | clk_divisor_reset | If ‘0’, reset clock divisor. |
| | [12:11] | | source | ‘00’ - reference VC, ‘01’ - gpio[0] all edges, ‘10’ gpio[0] rising edges, ‘11’ gpio[0] falling edges |
| | [13] | | clk_divisor_load_now | If set to ‘1’, loads adaptive module A’s divisor. |
| 71Ah | [15:0] | adapb1 | div_integer | Adaptive module A - integer divisor |
| 71Ch | [15:0] | adapb2 | div_fraction | Adaptive module A - fractional component of divisor |
| 740h | [4:0] | gpio_out_reg0 | gpio_0_output_select | See Table 32, “GPIO mux, Output Selection,” on page 82. |
| | [5] | | gpio_0_output_constant | Constant value of ‘00110’ (see Table 32) |
| | [6] | | gpio_0_output_enable | ‘0’ - tri-state, ‘1’ - output enabled. |
| | [12:8] | | gpio_1_output_select | see Table 32 |
| | [13] | | gpio_1_output_constant | Constant value of ‘00110’ (see Table 32) |
| | [14] | | gpio_1_output_enable | ‘0’ - tri-state, ‘1’ - output enabled. |
| 742h | [4:0] | gpio_out_reg1 | gpio_2_output_select | See Table 32, “GPIO mux, Output Selection,” on page 82. |
| | [5] | | gpio_2_output_constant | Constant value of ‘00110’ (of Table 32) |
| | [6] | | gpio_2_output_enable | ‘0’ - tri-state, ‘1’ - output enabled. |
| | [12:8] | | gpio_3_output_select | See Table 32, “GPIO mux, Output Selection,” on page 82. |
| | [13] | | gpio_3_output_constant | Constant value of ‘00110’ (of Table 32) |
| | [14] | | gpio_3_output_enable | ‘0’ - tri-state, ‘1’ - output enabled. |
| 744h | [4:0] | gpio_out_reg2 | gpio_4_output_select | See Table 32, “GPIO mux, Output Selection,” on page 82. |
| | [5] | | gpio_4_output_constant | Constant value of ‘00110’ (of Table 32) |
| | [6] | | gpio_4_output_enable | ‘0’ - tri-state, ‘1’ - output enabled. |
| | [12:8] | | gpio_5_output_select | See Table 32, “GPIO mux, Output Selection,” on page 82. |
| | [13] | | gpio_5_output_constant | Constant value of ‘00110’ (See Table 32 - on page 82). |
| | [14] | | gpio_5_output_enable | ‘0’ - tri-state, ‘1’ - output enabled. |
| 746h | [4:0] | gpio_out_reg3 | gpio_6_output_select | See Table 32, “GPIO mux, Output Selection,” on page 82. |
| | [5] | | gpio_6_output_constant | Constant value of ‘00110’ (see Table 32, “GPIO mux, Output Selection,” on page 82). |

**Table 30 - Clock Recovery Registers (continued)**

| Address | Bits | Register | Bits | Description |
|---|---|---|---|---|
| | [6] | | gpio_6_output_enable | ‘0’ - tri-state, ‘1’ - output enabled. |
| | [12:8] | | gpio_7_output_select | See Table 32, “GPIO mux, Output Selection,” on page 82. |
| | [13] | | gpio_7_output_constant | Constant value of ‘00110’ (of Table 32) |
| | [14] | | gpio_7_output_enable | ‘0’ - tri-state, ‘1’ - output enabled. |
| 748h | [4:0] | gpio_out_reg4 | ct_netref1_output_select | See Table 32, “GPIO mux, Output Selection,” on page 82. |
| | [5] | | ct_neterf1_output_constant | Constant value of ‘00110’ (See Table 32, “GPIO mux, Output Selection,” on page 82). |
| | [6] | | ct_netref1_output_enable | ‘0’ - tri-state, ‘1’ - output enabled. |
| | [12:8] | | ct_netref2_output_select | See Table 32, “GPIO mux, Output Selection,” on page 82. |
| | [13] | | ct_netref2_output_constant | Constant value of ‘00110’ (of Table 32) |
| | [14] | | ct_netref2_output_enable | ‘0’ - tri-state, ‘1’ - output enabled. |
| 802h | [1] | status0 | pointa_buf_overflow | Overflow occurred in the point A buffer. |
| | [2] | | pointb_buf_overflow | Overflow occurred in the point B buffer. |
| 804h | [1] | satus0_ie | pointa_buf_overflow | If ‘1’, pointa_buf_overflow will generate interrupt. |
| | [2] | | pointb_buf_overflow | If ‘1’, pointb_buf_overflow will generate interrupt. |
| 820h | [8:0] | pointa_manage | pointa_buffer_base_add | Bits 20:12 of the base address of the clock recovery point buffer. |
| | [11:9] | | pointa_buffer_size | see Table 31 |
| 822h | [13:0] | pointa_read | pointa_rpnt | CPU’s read pointer to the clock recovery point FIFO. |
| 824h | [13:0] | pointa_write | pointa_wont | Chip’s write pointer to the clock recovery point FIFO. |
| 828h | [8:0] | pointb_manage | pointb_buffer_base_add | Bits 20:12 of the base address of the clock recovery point buffer. |
| | [11:9] | | pointb_buffer_size | See Table 31, “Buffer Sizes,” on page 81. |
| 82Ah | [13:0] | pointb_read | pointb_rpnt | CPU’s read pointer to the clock recovery point FIFO. |
| 82Ch | [13:0] | pointb_write | pointb_wont | Chip’s write pointer to the clock recovery point FIFO. |

**Table 30 - Clock Recovery Registers (continued)**

| Register Value | Memory Allocated |
|---|---|
| 000 | 4K |
| 001 | 8K |
| 010 | 16K |
| 011 | 32K |
| 100 | 64K |
| 101 | 128K |
| others | reserved |

**Table 31 - Buffer Sizes**

| gpio_x_output_select | Signal |
|---|---|
| 00000 | ct_c8_a_in |
| 00001 | ct_c8_b_in |
| 00010 | ct_frame_a_in |
| 00011 | ct_frame_b_in |
| 00100 | ct_c8_selected |
| 00101 | ct_frame_selected |
| 00110 | output constant (“0”/“1”) |
| 00111 | ct_netref1_in |
| 01000 | ct_netref2_in |
| 01001 | gpio_in(0) |
| 01010 | gpio_in(1) |
| 01011 | gpio_in(2) |
| 01100 | gpio_in(3) |
| 01101 | gpio_in(4) |
| 01110 | gpio_in(5) |
| 01111 | gpio_in(6) |
| 10000 | gpio_in(7) |
| 10001 | adapa_ref |
| 10010 | adapb_ref |
| 10011 | clkrecov_pulse_a |
| 10100 | clkrecov_pulse_b |
| 10101 | mc_clock |
| 10110 | ct_mc_in |

**Table 32 - GPIO mux, Output Selection**

## 2.9 Silence Suppression

### 2.9.1 Overview

The Variable Bit Rate properties of the AAL2 standard permits the use of silent suppression. The MT90502 exploits this feature to enable optimal bandwidth usage for voice communications. The MT90502 has the capability to determine if an AAL2 CPS-Packet (transmitted or received) is considered silent. Two configurations can be employed for silent suppression: simple silence suppression and complex silence suppression. Simple silence suppression mode uses an external resource or the match & mask feature determines the silent status of the byte. In complex silence suppression mode the MT90502 determines the silent status on the current CPS-Packet. The MT90502 maintains a maximum channel capacity of 1023 channels with or without silence suppression enabled.

### 2.9.2 Simple Silent Suppression

#### 2.9.2.1 Silent Bit Indication

Simple Silent Suppression relies upon an external resource to perform the silent decision operation unless the match and mask method is employed. The result yielded from the external resource is conveyed to the MT90502 by a silent bit in the associated TSST (see Figure 50 on page 83). A one in silent bit indicates that the accompanying PCM or ADPCM sample is a silence. The position of silent bit is programmable through register 500h. Upon the reception of a complete CPS-Packet of silence, the CPS-Packet is considered silent and the process of generating an SID CPS-Packet is initialized.

The Noise Level field in SID packet will contain an idle code (no noise).

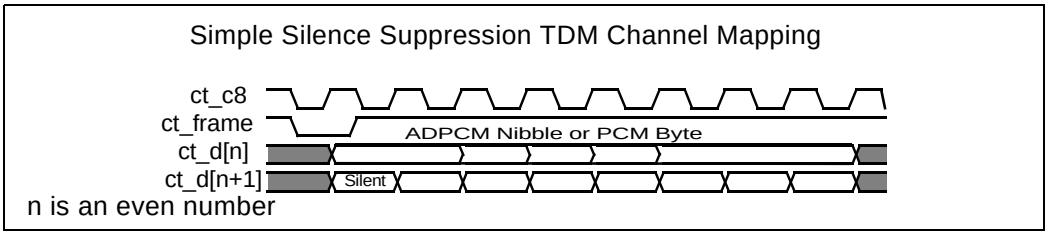



**Figure 50 - Simple Silent Suppression Stream Configuration**

#### 2.9.2.2 Last Byte Indication

This method is only available when TDM Format B is deployed. In this mode, external silence suppression engine shall make the decision whether a CPS-Packet is silence or voice, and indicate a silent packet by putting 00h on the last byte of that packet, together with PCM bit equal to 0 (see Figure 9 on page 27 and Figure 11 on page 29). MT90502 calculates the average energy of the packet based on the Tx PCM unsigned magnitudes received on the first TSST. Tx PCM magnitude given to MT90502 must be an unsigned value without DC offset. Upon detecting silent packet pattern at the end of the packet, the MT90502 will discard that packet and start silence period. Should a SID packet be sent out, MT90502 will insert the calculated energy level into SID byte.

#### 2.9.2.3 Match and Mask Determines Silence

The match and mask filtering operation operates in a similar method as the UTOPIA match and mask. There are two match and mask registers: ‘Silent Pattern Detection Register A’ and ‘Silent Pattern Detection Register B’ address (512h and 514h respectively). The registers are configured to compare selected bits of the PCM value. The mask field determines which bits to compare and the match field detects silent values. If the byte yields a 0 result from the match and mask filter, then the byte is regarded as silent. Upon the reception of a complete CPS-Packet of silence, the CPS-Packet is considered silent and the process of generating an SID CPS-Packet is initialized.

The Noise Level field in SID packet will contain an idle code (no noise).

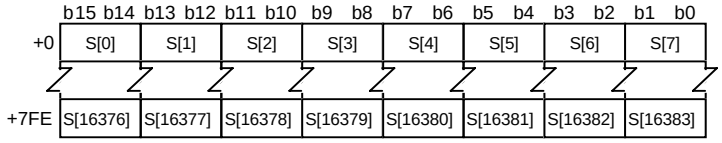
| | | |
|---|---|---|
| TDM Data Byte | 1 0 1 1 0 0 1 0 | 1 0 1 1 0 **1** 1 0 |
| Match Value | 1 0 1 1 0 0 1 0 | 1 0 1 1 0 **0** 1 0 |
| Match Result (1 = Mismatch) | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 **1** 0 0 |
| Mask Value | 1 1 1 1 1 1 1 1 | 1 1 1 1 1 **1** 1 1 |
| Mask Result (1 = mismatched cell) | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 **1** 0 0 |
| Result | Considered Silent | Not Silent |
| For each bit, result = (match XOR header) AND mask | | |

**Figure 51 - Silent Suppression Mask & Match Example**

### 2.9.3 Complex Silent Suppression

Complex silence suppression eliminates the external resource requirement to determine silent byte or silent packet. The MT90502 analyses both the remote (Rx direction) and local (Tx direction) signals and evaluates if silence is to be suppressed. The remote and local signals are processed to determine their average dB value over a CPS-Packet. The mean average dB of the CPS-Packet for both directions are employed to identify the transmission state as defined in the user configurable State Graph (Figure 52 on page 85).

**Silence Suppression State Table**




-Index = (((-Local Energy) *128)+(-Remote Energy)) where the local and Remote Energies range from 0 to -78 dBm0 inclusively.
Index refers to S[index]
-The S State is defined as ‘00’=voice; ‘01’=transition - maintain the status of the last non-transitional packet; ‘10’=silence; ‘11’=reserved.

**Typical Silence Suppression State Table:**

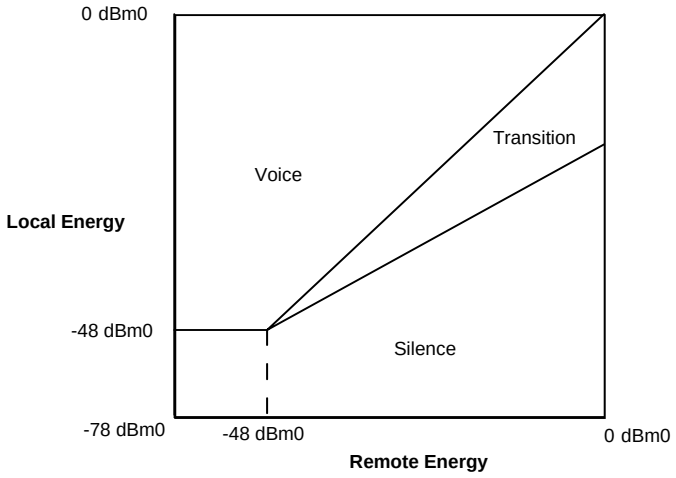



**Figure 52 - Complex Silent Suppression Configurable State Graph**

### 2.9.3.1 Complex Silent Suppression Operation

#### 2.9.3.1.1 PCM Law Table

PCM A-law and μ-law translation is supported by MT90502 on a per channel basis. For RX PCM channel, input law (from recieved CPS packet) and output law (sent on TDM bus) are controlled through RX TDM Control Memory Structures as shown in Figure 29, “RX Channel Association Memory (RX CAM),” on page 57.

For TX PCM channels, law translation is configured in PCM Law Table located at 1000h-11FEh.

| | b15 b14 | b13 b12 | b11 b10 | b9 b8 | b7 b6 | b5 b4 | b3 b2 | b1 b0 |
|---|---|---|---|---|---|---|---|---|
| 1000h | | | | | ch[0] | ch[1] | ch[2] | ch[3] |
| 1002h | | | | | ch[4] | ch[5] | ch[6] | ch[7] |
| | | | | | ... | ... | ... | ... |

ch[x]: Law translation for TX PCM channel number X.

00 - µ-law to µ-law

01 - µ-law to A-law

10 - A-law to µ-law

11 - No translation

**Figure 53 - PCM Law Table**

#### 2.9.3.1.2 DC Offset Calculation

The DC component is calculated in both remote and local voice PCM bytes. The MT90502 is able to perform an accumulation of the PCM linear values over a number of PCM samples (see Local and Remote DC Offset Figure 61 on page 91) in the reception direction. Accumulation of DC offset is enabled only when CPS-Packet Assembly Structure is in DC Calculation state. After obtaining DC offsets a short while after a call begins, the CPU is then required to change the Assembly Structure to Transition state, and write the DC Offset Correction into the Structure. After that, the Structure can be configured to Suppression state where suppression circuit is activated. No silence is suppressed in either DC calculation state or Transition state.

#### 2.9.3.1.3 Signal Energy Calculation

To obtain the signal energy, the linear PCM value with consideration of the DC offset correction is squared. The MT90502 sums all the byte energy values contained in a CPS-Packet and divides the result by the length of the CPS-Packet to obtain an average energy per byte. Then the MT90502 calculates the linear energy value into its decibel value.

The same process is employed with ADPCM; however, an external CODEC is required to convert the ADPCM to PCM. ADPCM silent suppression TDM streams are associated in groups of 4: for example, the ADPCM channel in transmission is mapped on stream 3, while the local PCM channel is mapped on stream 0, the ADPCM reception channel is mapped on stream 2, and the remote PCM channel on stream 1 (see Figure 54 and Figure 55).

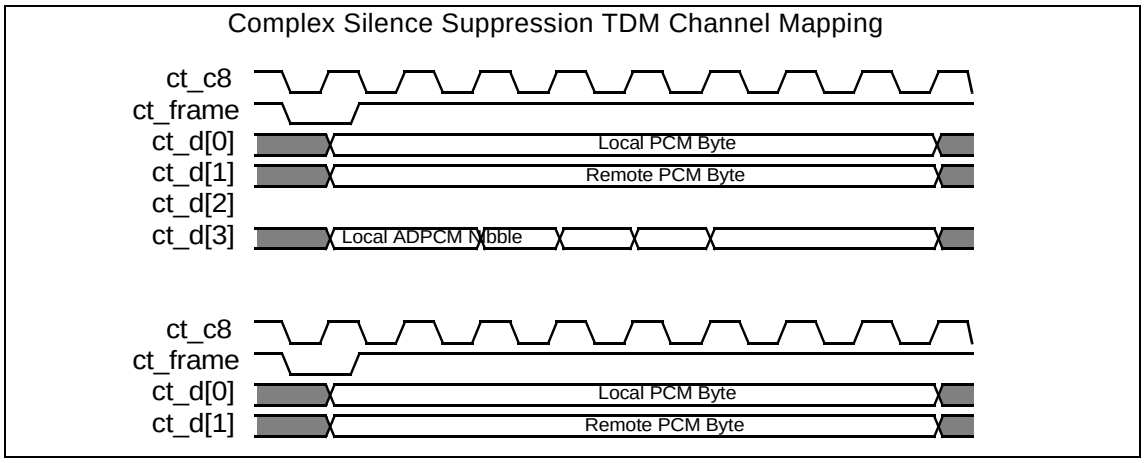



**Figure 54 - Complex Silent Suppression Stream Configuration**

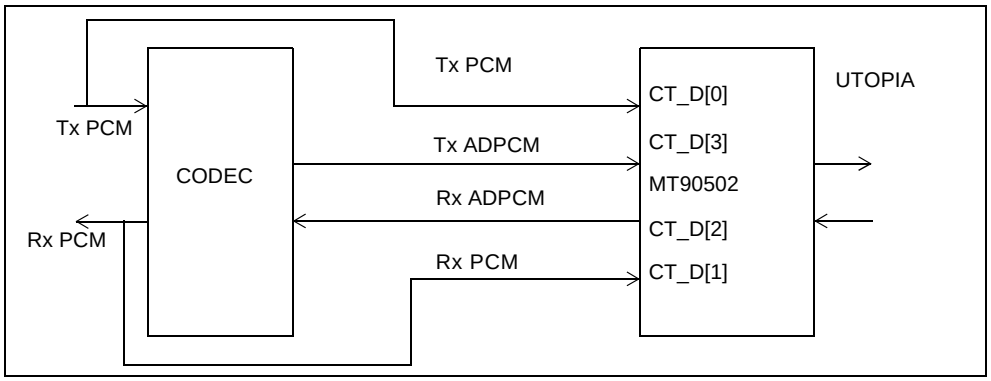



**Figure 55 - ADPCM Complex Silent Suppression Stream Configuration**

#### 2.9.3.2 CPS-Packet Silence State

The procedure is identical for both remote and local channels, as the silence suppression function requires information for both local and remote CPS-Packet mean values to consult the State Graph (see Figure 52) and determine the CPS-Packet status. Therefore, a channel on which complex silence suppression is enabled must have its transmit and receive direction streams mapped on consecutive stream and the same time slot (e.g. see Figure 54, “Complex Silent Suppression Stream Configuration,” on page 86). For example, the transmission $stream_{(odd)}$ 17, time slot 4 the associated reception stream is reception $stream_{(even)}$ 16 time slot 4.

##### 2.9.3.2.1 Silence Suppression State Table

To determine a CPS-Packet status (i.e. voice, transitional & silence), the TX TDM will reference the local and remote decibel energy levels in external memory. The table contains 2-bit values indicating a combination of local and remote energy levels. Note the Silent State Graph as shown in Figure 52.

The Silence Suppression State Table (4KB) is located in the variable structures of TX SSRAM determined by the ‘Table Offset[20:12]’ field in the CPS-Packet Final Assembly Structure (see Figure 63). The MT90502 can accommodate a different State Table for as many as 512 channels, as different channels may exhibit different line characteristics. The Table Offset field in the CPS-Packet Final Assembly Structure (shown in Figure 63) points to the base address of the Silence Suppression State Table for the associated channel.

##### 2.9.3.2.2 SID Transmission Operation

After the transmission of an SID, subsequent silent CPS-Packets are suppressed. However, the UUI sequence count is maintained. Therefore the reception device can determine the number of CPS-Packets suppressed. Upon the detection of silence the first suppressed CPS-Packet will be replaced by an SID packet and will contain the Local Energy determined by the CPS-Packet Assembly Structure, without consideration of the Local Energy Correction**.** Further SID CPS-Packets will be transmitted if the Local Energy is less than 3dB of the last transmitted SID energy level. Therefore, the energy level transmitted may gradually reduced over the silent period.

The ratio of silent packets to total CPS-Packets can be determined by consulting the CPS-Packet Final Assembly Structure, SID Count and CPS-Packet Count fields (see Figure 62 and Figure 63).

##### 2.9.3.2.3 SID Reception Operation

In the reception direction, SID CPS-Packets are employed to insert silent/comfort noise in Rx direction. The MT90502 is required to compensate for suppressed CPS-Packets employing the CPS-Packet loss compensation algorithm in the RX SAR (see Section 2.4.4, CPS-Packet Loss Compensation on page 52).

Therefore, it is a requirement to enable the CPS-Packet Loss Compensation employing the UUI sequence number counter in silence suppression mode.

When the RX SAR receives an SID CPS-Packet and in the CPS-Packet Disassembly Structure (see Figure 24 on page 47), LI0 field is configured for Silent Suppression the SID value will identify the Silent/Tone Buffer Number by referencing the SID Byte to Silence Buffer Structure (see Figure 56 on page 88). The SID energy level is mapped to the Silent/Tone Buffer Number in internal memory 1400h to 15FEh. For each SID byte, the Silence/Tone Buffer Number is stored in the SID Byte to Silence Buffer memory location. Different silence buffers must be used depending on the compression type. There are 5 entries for each SID byte, one per compression type:

- PCM 64kb/s
- ADPCM 40kb/s
- ADPCM 32kb/s
- ADPCM 24kb/s
- ADPCM 16kb/s

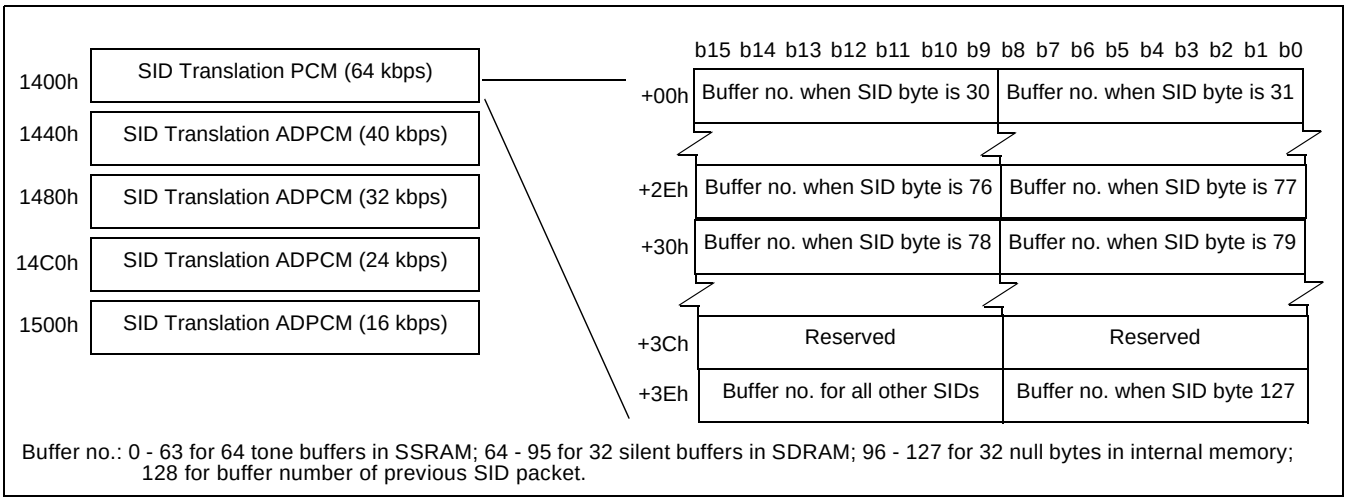
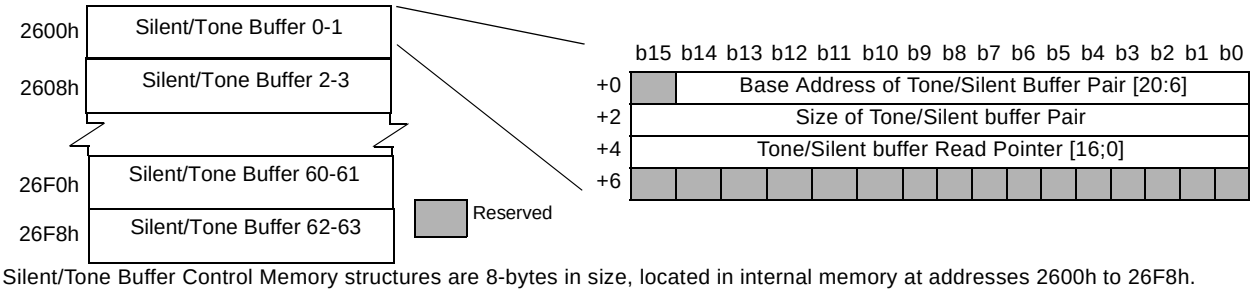



**Figure 56 - SID Byte to Silence Buffer Structure**

The Silent/Tone Buffer Number is written to the ‘Tone Number to Insert’ field located in the RX TDM Control Memory Structure (see Figure 31 on page 59) which identifies the ‘Silent/Tone Buffers’ (e.g. Figure 57 on page 89) in either SSRAM, SDRAM or null byte insertion, for the associated channel. The contents of the ‘Silent/Tone Buffers’ is inserted on the channel in place of the TDM Circular Buffer.

One byte from each silent noise buffer is read at the beginning of each frame and written into internal memory. Therefore a silent/tone value for each active silent channel is primed for the entire frame. The MT90502 can use up to 96 silent noise buffers. In addition, up to 32 null bytes can be used, augmenting the actual number of padding possibilities to 128. At the start of each frame, the MT90502 reads the address, size, and current read pointer of every silent noise buffer, and then reads the corresponding byte. It then updates the read pointer for the next frame and writes it back.




Silent/Tone Buffer Control Memory structures are 8-bytes in size, located in internal memory at addresses 2600h to 26F8h.

**Base Address of Tone/Silent Buffer Pair:** Points to a 64-byte boundary at which this silent buffer pair begins.

**Size of Tone/Silent Buffer Pair:** Indicates the size of this silent buffer pair.

0000h = invalid (this pair is not used)., 0001h = 1byte (per pattern), FFFFh = 65535-bytes (per pattern). Because this controls the size of 2 buffers, a full-sized pair will occupy 128K-bytes.

**Tone/Silent buffer Read Pointer:** Current Read Pointer to this silent buffer pair. This field must be set to 0 by software when initialised.

**Figure 57 - SSRAM Tone Buffer Control Memory**

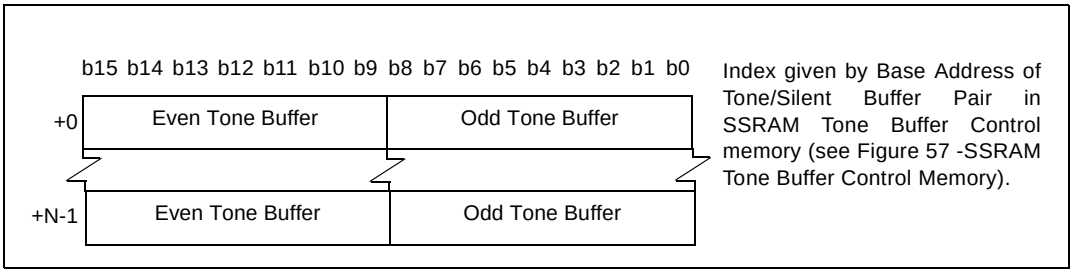



**Figure 58 - Silent Tone Buffer Pair in TX SSRAM**

32 null bytes are stored in two locations in internal Tone Data Buffer Memory. Both locations must have identical copies.

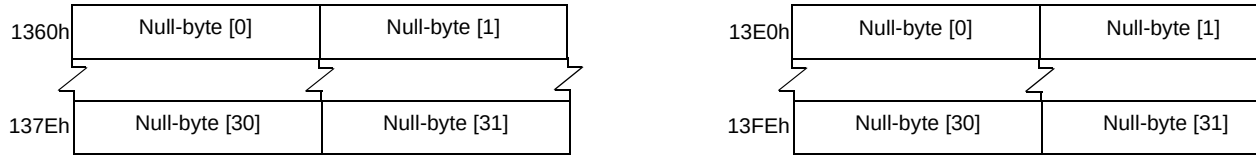



**Figure 59 - Null Bytes in Tone Data Buffer Memory**

| | | |
|---|---|---|
| Buffer[0] | Buffer [1] | Byte[0] |
| ... | ... | |
| Buffer[30] | Buffer[31] | |
| Buffer[0] | Buffer [1] | Byte[1] |
| ... | ... | |
| Buffer[30] | Buffer[31] | |
| ... | ... | ... |
| Buffer[0] | Buffer[1] | Byte [size - 1] |
| ... | ... | |
| Buffer[30] | Buffer[31] | |

**Figure 60 - 32 SDRAM Silence Buffers**

### 2.9.4 Voice/Silence Timer

When a PCM or ADPCM CPS-Packet is composed entirely of silent bytes, then it is tagged as being a silent CPS-Packet. It is then sent off to the CPS-Packet Final Assembly Structure to be treated. A silent CPS-Packet will not necessarily be discarded: the Final Assembly Structure contains 2 fields, the **Voice to Silence Timer** and the **Silence to Voice Time**r, which indicate how long a channel must be silent before it starts being suppressed, and how long a channel must be active before it starts being transmitted again. Usually, the **Silence to Voice Timer** is close to 0, so that voice is never cut off at the beginning of a talk spurt. However, the **Voice to Silence Timer** can be much longer, because it is not as important to start suppressing silence as soon as voice trails off: leaving a certain silent time makes the transition to silence suppression much smoother. Note that both Timers are defined in CPS-Packets, which is the most precision possible in this application.

To improve the Silence to Voice transition, another field called the **TX Delay** adds a constant delay of 0 to 4 CPS-Packets to the transmission: in other words, when the system must transmit a new CPS-Packet, it will transmit the one from 2 CPS-Packets ago. Thus, by using a **Silence to Voice Timer** of 2 CPS-Packets and a **TX Delay** of 2 CPS-Packets, the Silence to Voice transition will have zero loss, while ensuring that small spikes of noise do not cause the system to believe that continuous voice is being transmitted.

**PCM/ADPCM CPS-Packet Assembly Structure**
**(in Complex Silence Suppression Mode, dc calcuation state)**

| | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| +0 | | | | | | | | | | | Sub Phase | | | | Phase | |
| +2 | | | | *Local DC Offset[20:8]* | | | | | | | | | | | | |
| +4 | | | | *Remot DC Offset[20:8]* | | | | | | | | | | | | |
| +6 | *Local DC Off.[7:0]* | | | | | | | | | | *Rem. DC Off.[7:0]* | | | | | |
| +8 | # EDU | | | Comp | | | 1 | 0 | *Partial Byte Storage[7:1]* | | | | | | | 0 |

**PCM/ADPCM CPS-Packet Assembly Structure**
**(in Complex Silence Suppression Mode, transition state)**

| | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| +0 | | | | | | | | | | | Sub Phase | | | Phase | | |
| +2 | | | | | | | | | | | | | | | | |
| +4 | | | | | | | | | | | | | | | | |
| +6 | | | | | | | | | | | | | | | | |
| +8 | # EDU | | | Comp | | | 0 | 1 | *Partial Byte Storage[7:1]* | | | | | | | 0 |

**PCM/ADPCM CPS-Packet Assembly Structure**
**(in Complex Silence Suppression Mode, suppression state)**

| | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| +0 | | | | | | | | | | | Sub Phase | | | Phase | | |
| +2 | | | | *Local Energy Mantissa[7:0]* | | | | | | | *Local E. Exp[4:0]* | | | | | |
| +4 | | | | *Remote Energy Mantissa[7:0]* | | | | | | | *Remote E. Exp[4:0]* | | | | | |
| +6 | Local DC Correc.[7:0] | | | | | | | | | | Rem. DC Correc.[7:0] | | | | | |
| +8 | # EDU | | | Comp | | | 1 | 1 | *Partial Byte Storage[7:1]* | | | | | | | 0 |

Fields in *Italic* are used by Hardware only.

Fields in Plain are written to by the CPU/Software.

"00" state (addr+8[b9:b8]) is for simple silence suppression or no silence suppression. Structure shown in TDM Transmission section.

Reserved

**LocalDC Offset / Remote DC Offset**: Sum of all linear values of the PCM bytes received on the Local and Remote PCM channels (21-bits Twos-complement). This sum is sticky and will stay at -1048576 or 1048575 in case of overflow.

**Local DC Correction / Remote DC Correction**: 8 bit twos-complement value used to remove DC offset to the Linear values before being used to calculate the energy of the Local and Remote signals.

**Local / Remote Energy Mantissa / Exponent** : Field used by hardware to calculate the energy sum over a single packet. Exponent is defined as follows: “00001” = 1.0 to 1.99; “00010” = 2.0 to 4.98; etc. The highest bit of mantissa is always ‘1’. The units of this value is the square of the linearised PCM values inputs (-2047 to +2047).

**Figure 61 - Silent Suppression PCM/ADPCM CPS-Packet Assembly Structures**

**CPS-Packet Final Assembly Structure (Simple Silence Suppression)**

| | b15 | b14 | b13 | b12 | b11 | b10 | b9 | b8 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| +0 | VC Number[9:0] | | | | | | | | | | Seq I | | Pac Del Cnt [3:0] | | | |
| +2 | CID | | | | | | | | | | SUI | | Send UUI[3:0] | | | |
| +4 | S. to V. Timer | | | | Voice to SilenceTimer | | | | | | | | | | TX Delay | |
| +6 | | | | | # EDU | | | | | | | | | | | |
| +8 | | | | | | | | | | | | | | | | |
| +A | | | | | | | | | | | | | | | | |
| +C | | | | | | | | | | | | | | | | |
| +E | *SS Timer[8:0]* | | | | | | | | | *SS* | | | | | | |
| +10 | *CPSP0 Base [7:3]* | | | | *CPSP0 Code* | | | | *CPSP1 Base [7:3]* | | | | *CPSP1 Code* | | | |
| +12 | *CPSP2 Base [7:3]* | | | | *CPSP2 Code* | | | | *CPSP3 Base [7:3]* | | | | *CPSP3 Code* | | | |
| +14 | *SID Count[31:16]* | | | | | | | | | | | | | | | |
| +16 | *SID Count[15:0]* | | | | | | | | | | | | | | | |
| +18 | *CPS-Packet Byte Count[31:16]* | | | | | | | | | | | | | | | |
| 1A | *CPS-Packet Byte Count[15:0]* | | | | | | | | | | | | | | | |
| 1C | *CPS-Packet Count[31:16]* | | | | | | | | | | | | | | | |
| 1E | *CPS-Packet Count[15:0]* | | | | | | | | | | | | | | | |

PS-Packet Final Assembly structures are located in TX SSRAM at ddresses +1000h to +1FE0h, +2000h to +3FE0h, +4000h to +7FE0h and 8000h to +FFE0h for 128, 256, 512 and 1023 channels repsectively.

ach structure is 32-bytes in size.

ields in *Italic* are used by Hardware only.
ields in Plain are written to by the CPU/Software.

**Silence to Voice Timer** Time in CPS-Packets to change from silence to voice. 0 means all voice packets will be sent.
**Voice to Silence Timer** Time in CPS-Packets to change from voice to silence. 0 means all silent packets will not be sent.
**TX Delay** Delay in CPS-Packets that will be added by the Silence Suppression Algorithm to prevent loss of voice. Range is 0 to 4 CPS-packets.

**SS Timer**: Silence State Timer. Increments when the suppression indication from the TDM bus is active during a whole CPS-Packet and the SS state is cleared, until Voice to Silence Timer is reached. Also increments when the suppression indication from the TDM bus is inactive for any byte in a whole CPS-Packet and the SS state is set, until Silence to Voice Timer is reached. Whenever neither of the previous conditions is met, this timer will reset to 0. When the state counter reaches the Silent to Voice Timer or Voice to Silent Timer, the SS bit will toggle thus the supression state will change.
**SS**: Current Silence Suppression State. '0' = send voice; '1' = suppress.
**CPS-Packet 0-3 Base Add** base address of the start of the 4 previous CPS-packets in the TX Circular Buffer. 0 is most recent; 3 is oldest CPS-packet. Initialized to 00h at start-up by software.
**SID Count** Count of the number of SID CPS-Packets sent out. SID packets in this mode are only sent upon the transition from Voice to Silence.
**CPS-Packet Byte Count** Byte counter of all payload bytes sent (payload is defined as the LI+1 in each CPS-Packet sent). SID packets that are sent will count as one byte. SID packets that are not sent will be counted as 1 byte.
**CPS-Packet Count** CPS-Packet counter of all packets. All SID packets (might they have been sent or not) are counted in this field.

**Figure 62 - CPS-Packet Final Assembly Structure (Simple Silence Suppression)**

**CPS-Packet Final Assembly Structure (Complex Silence Suppression)**

| | b15 b14 b13 b12 b11 b10 b9 b8 | b7 b6 b5 b4 | b3 b2 b1 b0 |
|---|---|---|---|
| +0 | VC Number[9:0] | Seq I | Pack Del Cnt[3:0] |
| +2 | CID | SUI | Send UUI[3:0] |
| +4 | S. to V. Timer | Voice to SilenceTimer | TX Delay |
| +6 | Recent Period | # of EDU | Table Offset[20:12] |
| +8 | Local Energy Correction | | Remote Energy Correction |
| +A | *Max Local Energy(dBm0)* | | *Max Remote Energy(dBm0)* |
| +C | *Silent Padding Energy(dBm0)* | | *Recent Maximum Local Energy* |
| +E | *SS Timer[8:0]* | *SS* *GS* *SIS* | *Recent CPSP Cnt* |
| +10 | *CPSP0 Base Add[7:3]* *CPSP0 Code* | | *CPSP1 Base Add[7:3]* *CPSP1 Code* |
| +12 | *CPSP2 Base Add[7:3]* *CPSP2 Code* | | *CPSP3 Base Add[7:3]* *CPSP3 Code* |
| +14 | *SID Count[31:16]* | | |
| +16 | *SID Count[15:0]* | | |
| +18 | *CPS-Packet Byte Count[31:16]* | | |
| +1A | *CPS-Packet Byte Count[15:0]* | | |
| +1C | *CPS-Packet Count[31:16]* | | |
| +1E | *CPS-Packet Count[15:0]* | | |

Reserved

Fields in *Italic* are used by Hardware only.
Fields in Plain are written to by the CPU/Software.

CPS-Packet Final Assembly structures are located in TX SSRAM at addresses +1000h to +1FE0h, +2000h to +3FE0h, +4000h to +7FE0h and +8000h to +FFE0h for 128, 256, 512 and 1023 channels repsectively. Each structure is 32 bytes in size.

**CPSPx Code:** “000” = PCM; “001” = ADPCM40; “010” = ADPCM32; “011” = ADPCM24; “100” = ADPCM16; “101” = PCM byte misaligned EDU;others = reserved.
**Recent Period**: Number of CPS-Packets on which the maximum recent energy will be calculated. 0 means SID energy is re-calculated on each CPS-Packet. 15 means that the SID energy is recalculated every 16 packets.
**Table Offset[20:12]** Absolute position of the Silence Suppression State Table in the TX SSRAM to be used with this channel.
**Local Energy Correction / Remote Energy Correction** Signed two’s completement offset added to the Local/Remote Energies before indexing into the Table. This value is in dB. A value of 1 means a signal of -10 dB will be interpreted as -11 dB. Minimum value is -78 dB, Maximum value is 49 dB.
**Max Local / Remote Energy** Maximum Local / Remote Energy Monitored by hardware of a single CPS-Packet. 0 means 0 dBm0. 78 means -78 dBm0. Can be cleared by software to monitor over short periods.
**Recent CPSP Cnt, Silent Padding Energy & Recent Maximum Local Energy** Counter used to know when a new SID energy value has been obtained. This counter will increment until it matches the Recent Period field. Then it will be cleared. The CPS-Packet sent when this field is read and matches the Recent Period will be included in the Recent **Maximum Local Energy calculation**. Upon reading a Recent CPSP Cnt at 0, the field Recent Maximum Local Energy will be ignored (assumed to be -78 dB). The Silent Padding Energy will be set to the Maximum Local Energy upon terminal count when the resulting SS state is voice. When resulting SS state is silence, this operation is only performed when it will augument the value of the Silent Padding Energy (i.e. make the Silent Padding Energy more quiet). In this case, a SID CPS-Packet will be sent. When the first SID is sent, the Energy in the SID is the same as the one written back in the Silent Padding Energy.
**GS**: Graph State. ‘0’ = No voice suppression; ‘1’ = Suppress voice.
**SS Timer**: Silence State Timer. Increments when the Graph state and the SS bit are not equal, until Voice to Silence Timer or Silence to Voice timer are reached. Whenever the previous condition is not met, this timer will reset to 0. When the state counter reaches the Silent to Voice Timer or Voice to Silent Timer, the SS bit will toggle thus the supression state will change.
**SIS** Sid In Storage. When ‘1’, indicates that a SID packet would have been generated last time had the phase allowed it. This means a SID packet will be generated this time if the channel is deemed to remain silent.

**Figure 63 - CPS-Packet Final Assembly Structure (Complex Silence Suppression)**

## 2.10 HDLC

### 2.10.1 HDLC Overview

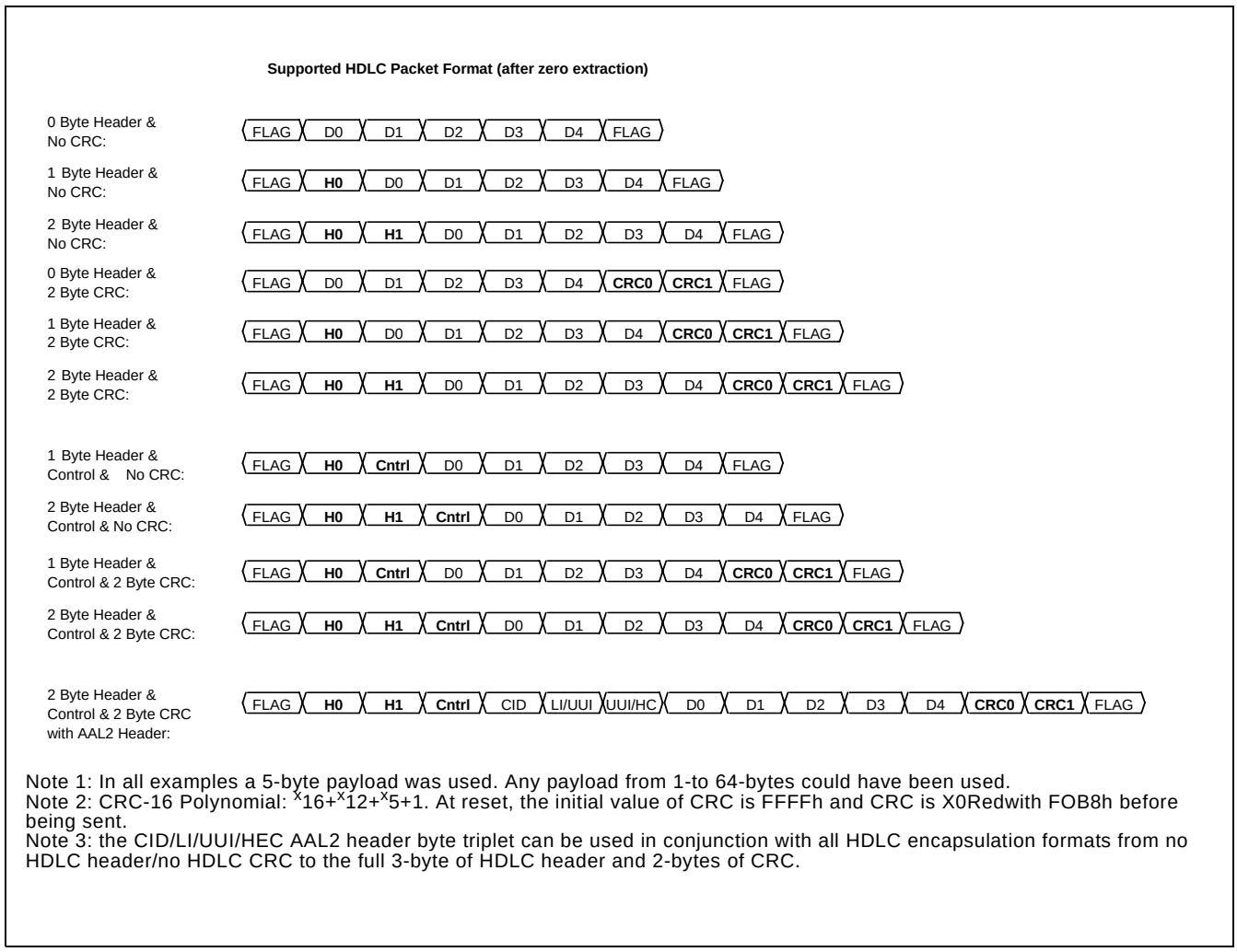
HDLC data format accommodates data that needs a larger bandwidth than PCM. Any number from 1 to 128 consecutive time slots on a single H.100/H.110 stream can carry data as one HDLC stream thus allowing a higher data transfer rate. Note that an HDLC stream must be contained within one frame and TX and RX HDLC streams carrying the same channel do not have to be of the same length.

### 2.10.2 HDLC Format

The MT90502 can accept or generate an HDLC header that contains 0, 1, or 2 address bytes, as well as a possible control byte that follows the flag. There is also an optional 16-bit CRC that may be added at the end of the packet. The different formats are illustrated in Figure 64 on page 95. The formats are specified for each stream using the “Header Type” and “CRC” fields in the HDLC CPS-Packet Assembly Structure (see Figure 13 on page 33 and Figure 14 on page 34). When using HDLC streams, the low byte of the address is used to select the channel number. All address bytes that are not used to indicate a channel number are discarded by the TX TDM module. The payload gamut of an HDLC CPS-Packet is 1 to 64-bytes.

Data is accumulated until a flag is received signifying the end of an HDLC packet. The payload will be processed according to the HDLC type. The MT90502 supports 2 types of HDLC data over the TDM bus: bit-wide form and byte-wide form (registers 400h and 500h).

Table 33, “HDLC Packet Formats and Header Types,” on page 96 shows the supported HDLC package formats with corresponding header types as shown in Figure 13. The CRC selection is also made according to Figure 13. Using packaing_type (500h bit-3) enables “Raw” AAL2 CPS-Packets to be sent, as described in Section 2.2.3.4. If there are not multiple HDLC channels in a HDLC stream then SCS will not be used and the address bytes will be discarded. If SCS is used then the address bytes will be used along with the HDLC stream number to calculate the HDLC channel number.




Note 1: In all examples a 5-byte payload was used. Any payload from 1-to 64-bytes could have been used.
Note 2: CRC-16 Polynomial: $^{x}16+^{x}12+^{x}5+1$. At reset, the initial value of CRC is FFFFh and CRC is X0Redwith FOB8h before being sent.
Note 3: the CID/LI/UUI/HEC AAL2 header byte triplet can be used in conjunction with all HDLC encapsulation formats from no HDLC header/no HDLC CRC to the full 3-byte of HDLC header and 2-bytes of CRC.

**Figure 64 - Supported HDLC Formats**

| Packet Format | Header Type | CRC | SCS | 500h [3] | Address Data |
|---|---|---|---|---|---|
| 0-Byte Header & No CRC | 0000 | 0 | No | 0 | - |
| 1-Byte Header & No CRC | 0001 | 0 | No | 0 | Discarded |
| 2-Byte Header & No CRC | 0010 | 0 | No | 0 | Discarded |
| 0-Byte Header & 2-Byte CRC | 0000 | 1 | No | 0 | - |
| 1-Byte Header & 2-Byte CRC | 0001 | 1 | No | 0 | Discarded |
| 2-Byte Header & 2-Byte CRC | 0010 | 1 | No | 0 | Discarded |
| 1-Byte Header & Control & No CRC | 0011 | 0 | No | 0 | Discarded |
| 2-Byte Header & Control & No CRC | 0100 | 0 | No | 0 | Discarded |
| 1-Byte Header & Control & 2-Byte CRC | 0011 | 1 | No | 0 | Discarded |
| 2-Byte Header & Control & 2-Byte CRC | 0100 | 1 | No | 0 | Discarded |
| 1-Byte Header & No CRC | 0101 | 0 | Yes | 0 | Used |
| 2-Byte Header & No CRC | 0110 | 0 | Yes | 0 | Used |
| 1-Byte Header & 2-Byte CRC | 0101 | 1 | Yes | 0 | Used |
| 2-Byte Header & 2-Byte CRC | 0110 | 1 | Yes | 0 | Used |
| 1-Byte Header & Control & No CRC | 0111 | 0 | Yes | 0 | Used |
| 2-Byte Header & Control & No CRC | 1000 | 0 | Yes | 0 | Used |
| 1-Byte Header & Control & 2-Byte CRC | 0111 | 1 | Yes | 0 | Used |
| 2-Byte Header & Control & 2-Byte CRC | 1000 | 1 | Yes | 0 | Used |
| 0-Byte Header & No CRC with AAL2 header | 0000 | 0 | No | 1 | - |
| 1-Byte Header & No CRC with AAL2 header | 0001 | 0 | No | 1 | Discarded |
| 2-Byte Header & No CRC with AAL2 header | 0010 | 0 | No | 1 | Discarded |
| 0-Byte Header & 2-Byte CRC with AAL2 header | 0000 | 1 | No | 1 | - |
| 1-Byte Header & 2-Byte CRC with AAL2 header | 0001 | 1 | No | 1 | Discarded |
| 2-Byte Header & 2-Byte CRC with AAL2 header | 0010 | 1 | No | 1 | Discarded |
| 1-Byte Header & Control & No CRC with AAL2 header | 0011 | 0 | No | 1 | Discarded |
| 2-Byte Header & Control & No CRC with AAL2 header | 0100 | 0 | No | 1 | Discarded |
| 1-Byte Header & Control & 2-Byte CRC with AAL2 header | 0011 | 1 | No | 1 | Discarded |
| 2-Byte Header & Control & 2-Byte CRC | 0100 | 1 | No | 1 | Discarded |
| 1-Byte Header & No CRC with AAL2 header | 0101 | 0 | Yes | 1 | Used |
| 2-Byte Header & No CRC with AAL2 header | 0110 | 0 | Yes | 1 | Used |
| 1-Byte Header & 2-Byte CRC with AAL2 header | 0101 | 1 | Yes | 1 | Used |
| 2-Byte Header & 2-Byte CRC with AAL2 header | 0110 | 1 | Yes | 1 | Used |
| 1-Byte Header & Control & No CRC with AAL2 header | 0111 | 0 | Yes | 1 | Used |
| 2-Byte Header & Control & No CRC with AAL2 header | 1000 | 0 | Yes | 1 | Used |
| 1-Byte Header & Control & 2-Byte CRC with AAL2 header | 0111 | 1 | Yes | 1 | Used |
| 2-Byte Header & Control & 2-Byte CRC with AAL2 header | 1000 | 1 | Yes | 1 | Used |

**Table 33 - HDLC Packet Formats and Header Types**

### 2.10.3 HDLC Bit-Wise Format

Bit-wise means that every bit of HDLC data coming from the H.100/H.110 bus is examined. A control flag of 7Eh ("01111110") is used to signify the start and end of a packet. When using this form of HDLC, each HDLC packet must begin with a flag and end with a flag, although a single flag may represent both the end of a packet and the beginning of another. A '0' is inserted after every 5 '1's of incoming data (called zero insertion) to differentiate the control flag and data. If neither flags nor data are being transmitted onto the bus, the idle code is transmitted. The idle code is an endless string of '1's. Note that a valid idle code must be at least 7-bits long (7 '1's).

### 2.10.4 HDLC Byte-Wise Format

Byte-wise HDLC format also employs "01111110" (7Eh) as a control flag. A 7Eh payload value is replaced by two bytes - 7Dh and 5Eh, while a 7Dh payload value is replaced by two bytes - 7Dh and 5Dh. A single flag may represent both the end of a packet and the beginning of another. This flag is put into the TX circular buffer with the data. When no data is being transmitted onto the bus, the flag character is sent repeatedly until data is transmitted again.

## 2.11 Memory

### 2.11.1 Memory Map

The location of the absolute starting and ending addresses of the internal and external memories are shown in Table 34. The complete set of internal registers is listed in Section 3, “Register List,” on page 107. The beginning and ending addresses of the various structure spaces in SSRAM and SDRAM are listed in Section 2.11.2, “Memory Structures,” on page 98.

| Start Address | End Address | Name |
|---|---|---|
| 0100h | 01FEh | cpureg |
| 0200h | 02FEh | mainreg |
| 0300h | 03FEh | txreg |
| 0400h | 04FEh | rxreg |
| 0500h | 05FEh | txtdmreg |
| 0600h | 06FEh | utoreg |
| 0700h | 07FEh | h100reg |
| 0800h | 08FEh | miscreg |
| 0900h | 09FEh | rxtdmreg |
| 1000h | 11FEh | PCM Law Table |
| 1300h | 13FEh | Tone Data Buffer Memory |
| 1400h | 15FEh | SID Byte to Silence Buffer Memory |
| 2000h | 20FEh | TX SAR Input FIFO |
| 2100h | 21FEh | AAL0 Input FIFO |
| 2200h | 22FEh | UTOPIA Port A Input FIFO |
| 2300h | 23FEh | UTOPIA Port B Input FIFO |
| 2400h | 24FEh | UTOPIA Port C Input FIFO |
| 2600h | 26FEh | Tone Buffer Control Memory |

**Table 34 - MT90502 Memory Map**

| Start Address | End Address | Name |
|---|---|---|
| 2800h | 2BFEh | UTOPIA Port A Output FIFO |
| 2C00h | 2FFEh | UTOPIA Port B Output FIFO |
| 3000h | 3FFEh | RX SAR Output FIFO |
| 4000h | 5FFEh | TX Channel Association Memory |
| 6000h | 7FFEh | RX Channel Association Memory |
| 8000h | BFFEh | TX TDM Control Memory |
| C000h | FFFEh | RX TDM Control Memory |
| 10000h | 103FEh | UTOPIA Port C Output FIFO |
| 400000h | 4FFFFEh | SSRAM bank A |
| 600000h | 7FFFFEh | SSRAM bank B |
| 2000000h | 2FFFFFEh | SDRAM bank A |
| 3000000h | 3FFFFFEh | SDRAM bank B |

**Table 34 - MT90502 Memory Map (continued)**

### 2.11.2 Memory Structures

There are two types of memory structures in the MT90502: fixed structures and variable structures. Fixed structures are of a defined size and base address, and are mapped according to the **sar_capacity** register field (register 240h), which determines the total channel capacity of the MT90502. The one exception is the RX Circular Buffers, whose size is determined by the **rx_circular_buffer_size** field (register 240h). The size of variable structures can be configured individually in registers.

When two memory banks exist, TX memory employs memory controller A and memory interface pins A, while RX memory uses memory controller B and memory interface pins B. When one memory bank exists, only memory controller A and memory interface pins A are used.

TX SSRAM fixed structures are (with sizes in bytes for 128, 256, 512, and 1023 channels)

- AAL2 VC structures (4K, 8K, 16K, 32K)
- CPS-Packet Final Assembly structures (4K, 8K, 16K, 32K)
- Cell Assembly Event Queue (512, 1K, 2K, 4K)
- CPS-Packet Circular Buffers (32K, 64K, 128K, 256K)

TX SSRAM variable structures are

- Clock Recovery Point Buffers (2 buffers the same size, either 4K, 8K, 16K, 32K, 64K, or 128 KB each)
- SSRAM Silence Buffers (up to 64, each pair can have a length of 0 to 64 KB per buffer)
- Silent Suppression Look-up Tables

RX SSRAM fixed structures are (with sizes in bytes for 128, 256, 512, and 1023 channels)

- AAL2 VC structures (16K, 32K, 64K, 128K)
- CPS-Packet Descriptor Queue (8K, 16K, 32K, 64K)
- CPS-Packet Disassembly structures (4K, 8K, 16K, 32K)
- CPS-Packet Descriptor Queue Pointers (512, 1K, 2K, 4K)
- CPS-Packet Circular Buffers (1 for each channel, all the same size 256, 512, or 1K bytes).

RX SSRAM variable structures are

- Data (AALO) cell FIFO (one FIFO: 4K, 8K, 16K, 32K, 64K, or 128K bytes)
- Error Report structure FIFO (one FIFO: 4K, 8K, 16K, 32K, 64K, or 128K bytes)
- CPU CPS-Packet buffer (one buffer: 16K, 32K, 64K, or 128K bytes)

TX SDRAM fixed structures are (with sizes in bytes for 128, 256, 512, and 1023 channels)

- CPS-Packet Descriptor Queue (512K, 1M, 2M, 4M)

TX SDRAM variable structures are

- Look-Up Tables[1] (3 tables, each can have a size of 1M, 2M, 4M, or 8 MB)
- SDRAM Silent Buffers (32 buffers the same size, either 0, 16K, 32K, or 64K bytes each)

RX SDRAM fixed structures are (with sizes in bytes for 128, 256, 512, and 1023 channels)

- RX CID table (1K, 2K, 4K, 8K)

There are no variable RX SDRAM structures.

1. All three LUTs can have the same base address. The sizes and base addresses of the LUTs are determined in registers 620h, 640h, and 660h.

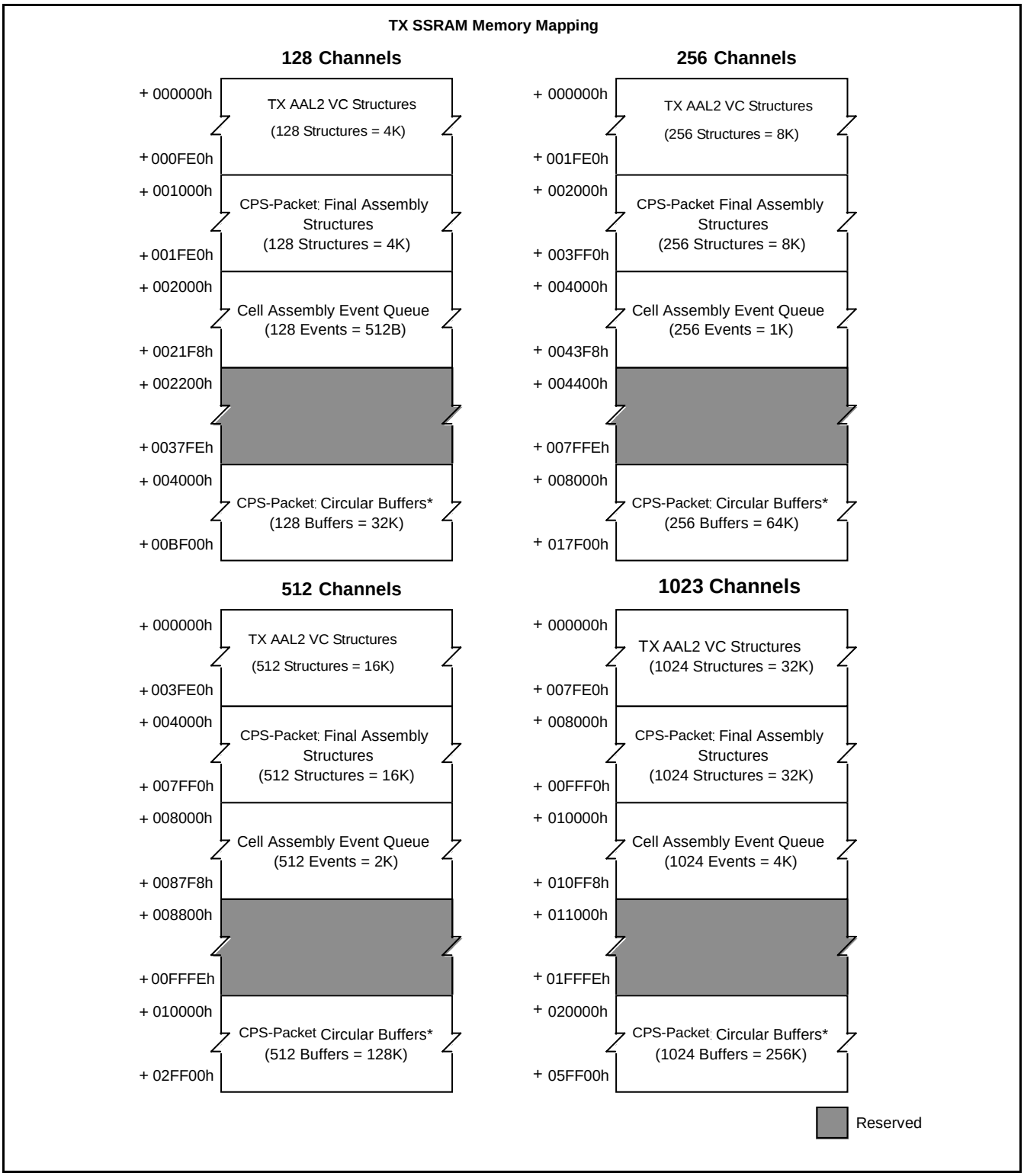



**Figure 65 - TX SSRAM Memory Mapping for Fixed Structures**

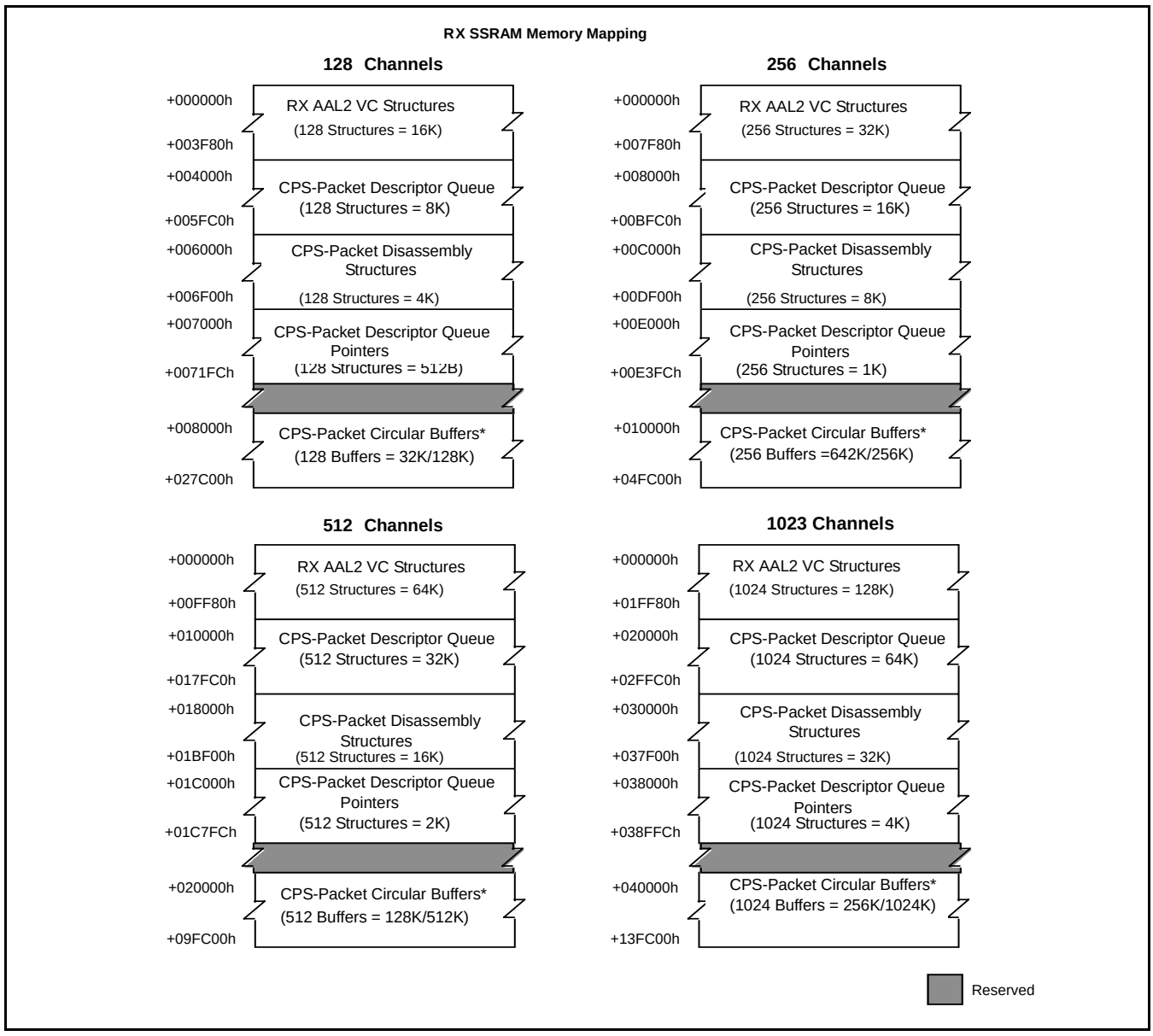



**Figure 66 - RX SSRAM Memory Mapping for Fixed Structures**

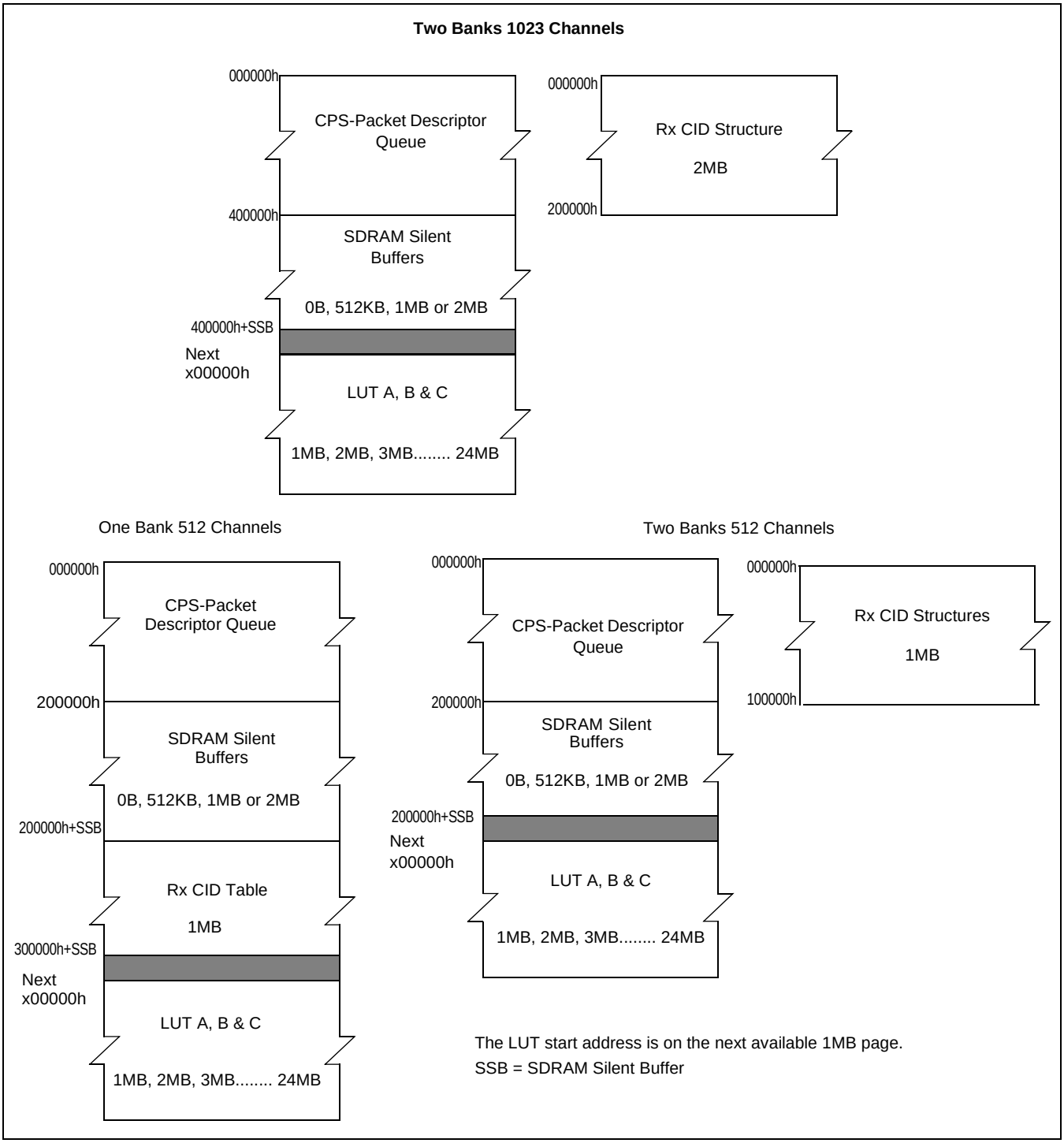



**Figure 67 - SDRAM Memory Map for 512 & 1023 Channels**

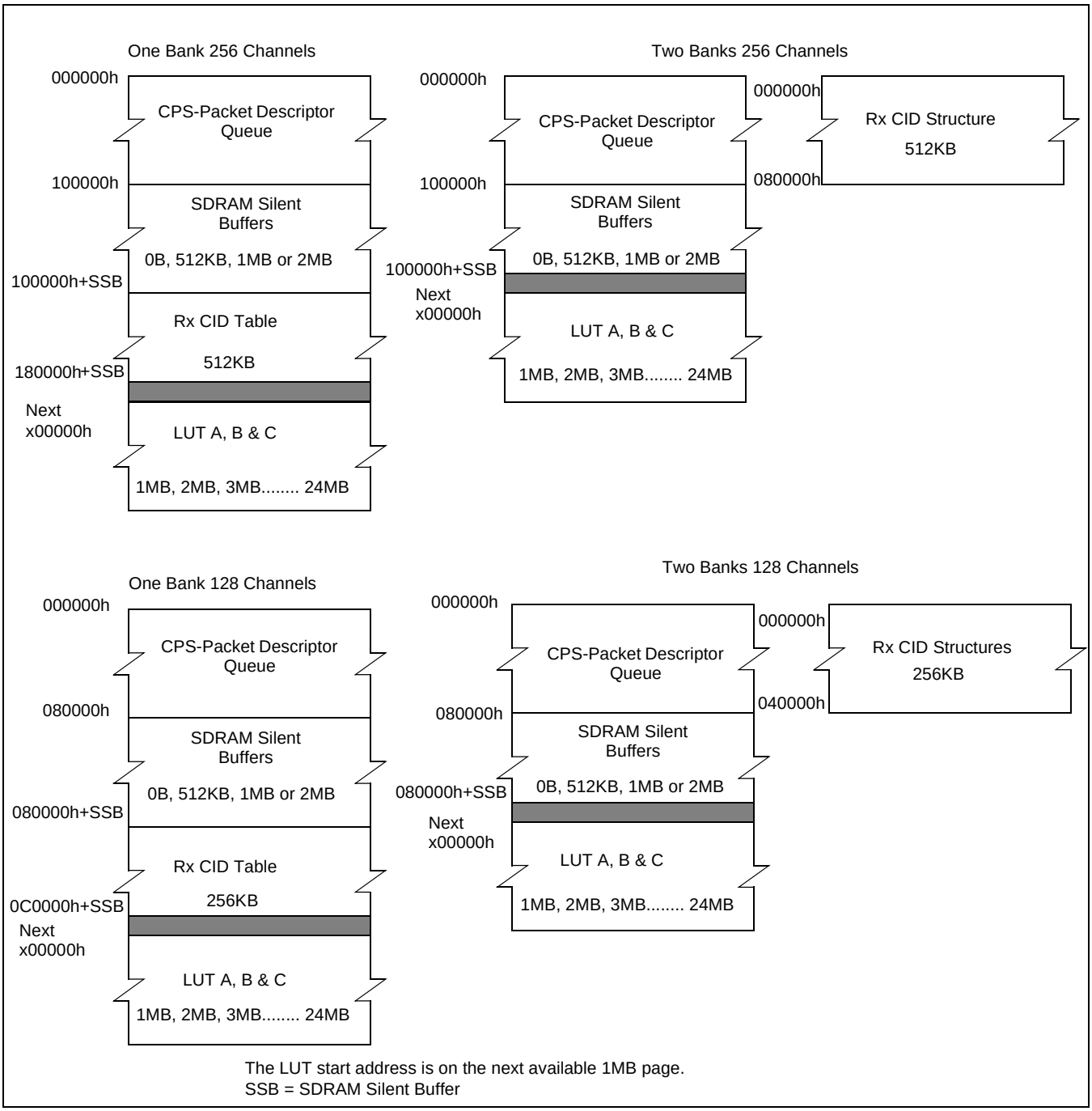



**Figure 68 - SDRAM Memory Map for 128 and 256 Channels**

### 2.11.3 Mem_Clk and Upclk

The memory clock is supplied to the MT90502 by an external source. The memories are connected to an external clock source which also must be coupled to the MT90502's mem_clk pin. A frequency of 60 MHz for mem_clk is recommended. Frequencies above 60 MHz will not increase performance, but will increase power consumption. At lower clock frequencies, the MT90502 may not be able to operate at its full SAR capacity.

Upclk is used by CPU interface circuit. Its frequency should not be less than half of mem_clk. There is no relationship between upclk to mem_clk.

### 2.11.4 Memory Controller

#### 2.11.4.1 Overview

The MT90502 uses up to 4 external SSRAM chips and 2 external SDRAM chips on 2 banks for its memory requirements. SSRAM chips can be 128KByte, 256KByte, 512KByte or 1MByte in size. The size of SDRAM chips is either 4Mx16 or 8Mx16. The address and data pins of the memories are shared on either bank (e.g. SDRAM A and SSRAM A share their pins). For applications of 512 channels or less, it is possible to use bank A only. Both banks must be used if sar_capacity is set to 1023.

If only 1 bank is employed, TX SSRAM starts from the beginning of bank A, which is 400000h. RX SSRAM starts from rx_base_address programmed in register 242h. If using two banks, however, bank A is always TX SSRAM and bank B is RX SSRAM.

A memory controller is used to multiplex the accesses required of these memories. This dual memory controller grants the memory bus to the various agents within the chip in order of urgency, using a priority algorithm, and transforms the memory accesses into the correct pin signals.

#### 2.11.4.2 Functionality

The memory controller is responsible for generating even parity on the parity pins of the memories and detecting that the parity is correctly received when reading data from the memory. The MT90502 calculates even parity on all address bits and data bits used to generate each access. When reading from the memory, it performs the same calculation in the opposite direction. Parity errors are reported to register 202h. To render parity generation and detection, configurable masks can be employed to calculate parity on certain bits (registers 230h, 232h and 234h).

Parity is calculated on all locations in memory except for the RX circular buffers, in which the parity bits are used for underrun information. It is possible to override this and use parity on these circular buffers through control bits in register 230h.

The controller ensures that the SDRAMs are refreshed often enough to avoid corrupt data. A limit (register 254h) exists for how many refresh periods behind the SDRAM can be before a status error is generated. SDRAM is configurable in registers 250h-25Ah.

### 2.11.5 Initializing SSRAM and SDRAM

The SSRAM parity generation and checking can be configured by programming registers 230h mem_parity0, 232h mem_parity1 and 234h mem_parity2. The SSRAM sizes can be configured by programming registers 240h mem_conf0 and 242h mem_conf1 (not programmed if using bank B for RX).

The SDRAM can be configured by a series of register writes. Pre-charge all the SDRAM banks, program the mode, perform a CBR refresh (twice), enable the SDRAM and then set the SDRAM for normal operation.

### 2.11.6 Memory Configuration

SSRAM memory chips must be ZBT (zero-bus turnaround) and must be pipelined. SDRAM memory chip must have A12-A13 as bank select pins, A0-A11 as row address and A0-A7/A8 as column address.

Typical application circuits for Bank A SSRAM and Bank A SDRAM are shown in Figure 69 on page 105 and Figure 70 on page 106.

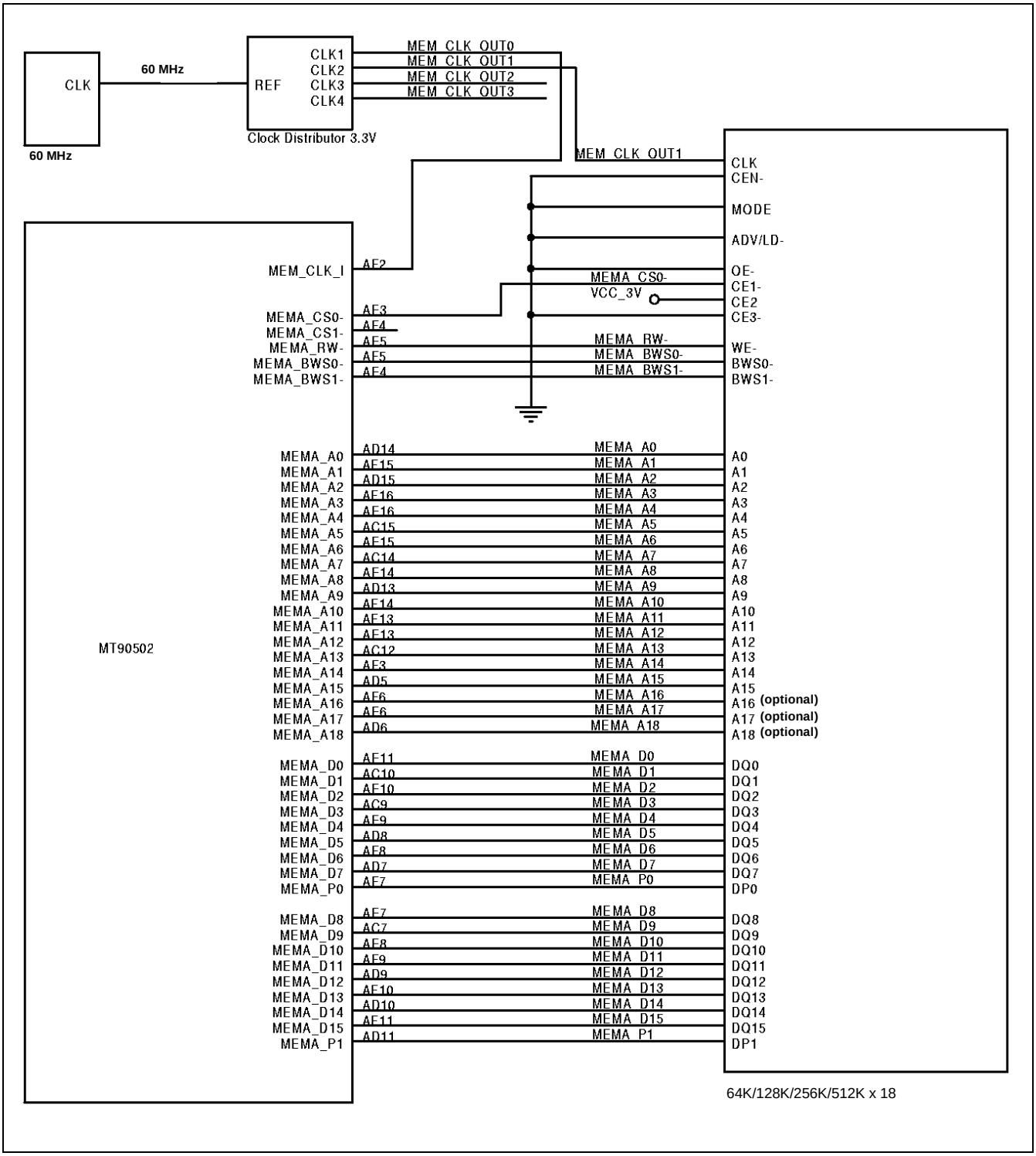



**Figure 69 - Typical SSRAM Application Circuit**

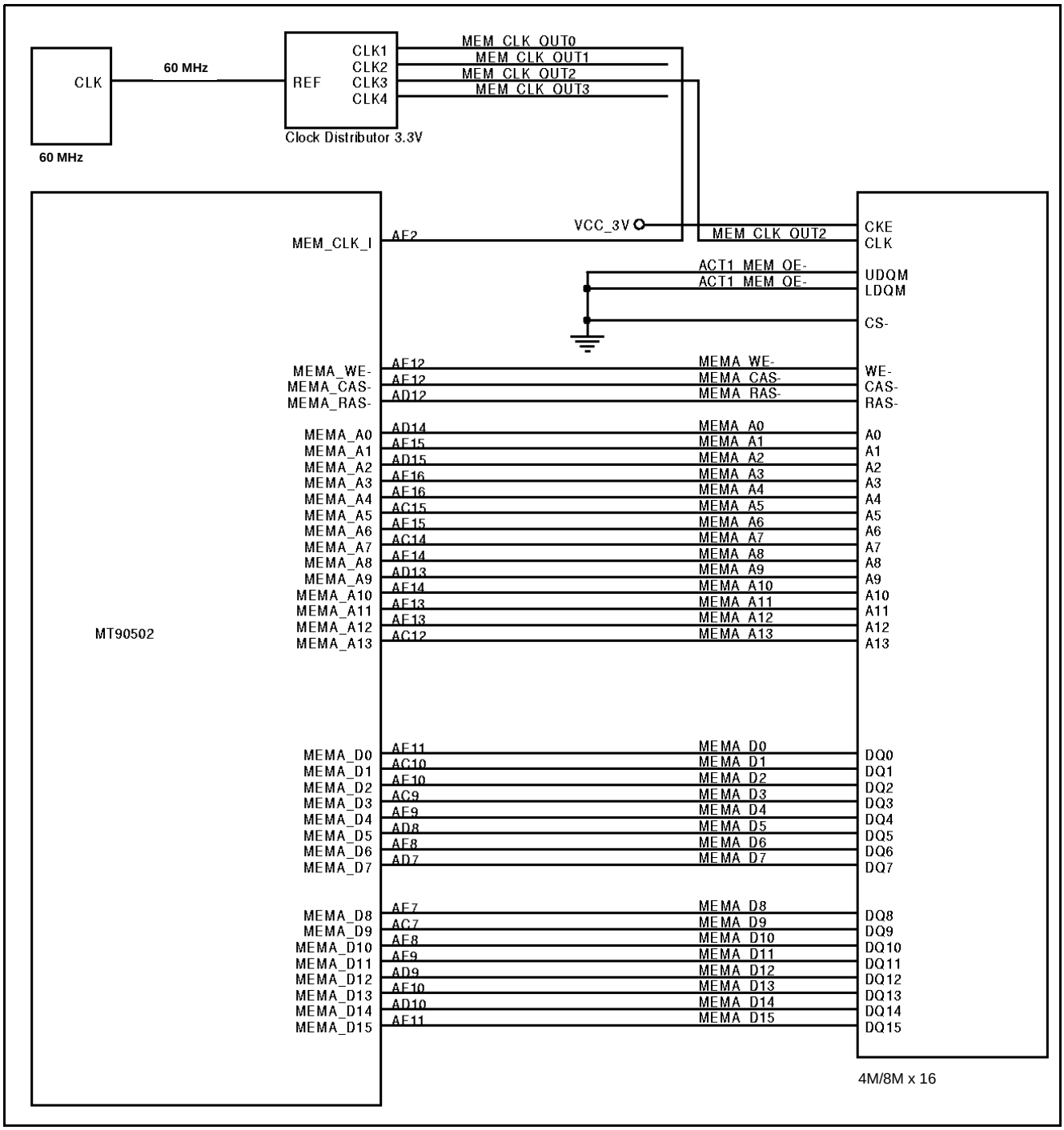



**Figure 70 - Typical SDRAM Application Circuit**

# 3.0 Register List

## 3.1 CPU Register

**Address: 100h**
**Label: control**
**Reset Value: 0000**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| sreset | 0 | RW | Active low software reset. Resets the whole chip except the CPU interface. Once low, all registers will be cleared except for CPU registers and Main Registers. |
| low_latency_cpu_accesses | 1 | RW | When '1', no caching will be done in the CPU Interface, thus guaranteeing a higher average access time, but a lower worst case access time. |
| reserved | 14:2 | RW | Reserved. Must always be “0000_0000_0000_0” |
| test_status | 15 | TS | Reserved. Must always be “0”. |

**Table 35 - CPU Control Register**

**Address: 102h**
**Label: status0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| reserved | 2:0 | ROL | Reserved. Always read as “000” |
| internal_read_timeout | 3 | ROL | Internal device time-out. Generally occurs when a clock is missing or misbehaving. |
| cpu_read_done | 4 | ROL | This bit is set when a burst of reads is completed. This bit may be used to generate an interrupt after a large read burst has completed (in indirection). For small read bursts, it is not very useful since there is not enough time for the CPU to do anything between the time it starts the read and the time that the read ends. |
| reserved | 15:5 | ROL | Reserved. Always read as “0000_0000_000” |

**Table 36 - CPU Status Register**

**Address: 104h**
**Label: status0_ie**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| reserved | 2:0 | RO | Reserved. Always read as “000” |
| internal_read_timeout_ie | 3 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| cpu_read_done_ie | 4 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| reserved | 15:5 | RO | Reserved. Always read as “0000_0000_000” |

**Table 37 - CPU Interrupt Enable Register**

**Address: 108h**
**Label: mem_clk_freq**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| reserved | 3:0 | RO | Reserved |
| mem_clk_freq_integer | 10:4 | RW | Frequency of mem_clk in MHz. |
| reserved | 15:11 | RW | Reserved. Must always be “0000” |

**Table 38 - Mem_clk Frequency Control Register**

**Address: 10Ah**
**Label: upclk_freq**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| reserved | 3:0 | RO | Reserved |
| upclk_freq_integer | 10:4 | RW | Frequency of upclk in MHz. |
| reserved | 15:11 | RW | Reserved. Must always be “0000” |

**Table 39 - Upclk Frequency Control Register**

**Address: 10Ch**
**Label: led0**
**Reset Value: 007Fh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| led_flash_freq[8:0] | 8:0 | RW | Determines the time in ms that the LEDs will be turned off to indicate link activity. |
| led_test_mode | 9 | RW | If '0', the LED Flashing time will be determined in ms. If '1', the LED Flashing time will be determined in us. |
| reserved | 15:10 | RW | Reserved. Must always be “0000_00” |

**Table 40 - LED Timing Control Register**

**Address: 16Ch**
**Label: fastclk_pll_conf0**
**Reset Value: 0642h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| reserved | 0 | WO | Reserved. Always read as “0” |
| pll_div_x | 3:1 | WO | Selects the division of the reference before being fed to the PLL. The reference is either upclk or mem_clk. |
| pll_div_y | 6:4 | WO | Selects the division of the feedback before being fed to the PLL. |

**Table 41 - Fast Clock PLL Configuration Register 0**

| | | | |
|---|---|---|---|
| pll_bypass | 7 | WO | When '1', mem_clk or upclk are passed directly (without being divided) to fast_clk. |
| pll_source | 8 | WO | 0' = upclk is PLL source; '1' = mem_clk is PLL source. |
| nreset_pll_async | 9 | WO | Active low PLL nreset. This value resets the PLL's clock divisors. |
| reserved | 10 | WO | Reserved. Must write “1”. |
| reserved | 15:11 | WO | Reserved. Always read as “0000_0” |

**Table 41 - Fast Clock PLL Configuration Register 0 (continued)**

**Address: 172h**
**Label: fastclk_pll_conf1**
**Reset Value: 09FCh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| pll_vcod | 1:0 | RW | Output frequency range selection bits = “00” |
| pll_pd | 2 | RW | ‘0’ for normal operation. Power Down = ‘1’. |
| pll_div | 7:3 | RW | Divider = “11111” |
| pll_syncen | 8 | RW | Forces PLL into a clock synchronisation mode = ‘1’ |
| pll_chp | 13:9 | RW | Charge Pump settings = “01000” |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 42 - Fast Clock PLL Configuration Register 1**

**Address: 174h**
**Label: h100pll_conf0**
**Reset Value: 07FEh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| H100pll_vcod | 1:0 | RW | Output frequency range selection bits = “10” |
| H100pll_pd | 2 | RW | ‘0’ for normal operation. Power Down = ‘1’. |
| H100pll_div | 7:3 | RW | Divider = “11111” |
| H100pll_syncen | 8 | RW | Forces PLL into a clock synchronisation mode = ‘1’ |
| H100pll_chp | 13:9 | RW | Charge Pump settings = “00011” |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 43 - H100/H110 PLL Configuration Register 0**

**Address: 17Eh**
**Label: chip_and_revision**
**Reset Value: 0101h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| chip_id[7:0] | 7:0 | RO | Chip ID = 01h. |
| rev_id[7:0] | 15:8 | RO | Revision ID = 01h. |

**Table 44 - ID Register**

### 3.2 Main Registers

**Address: 202h**
**Label: status0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mema_sdram_parity_error0 | 0 | ROL | Parity error on the high byte of bank A SDRAM. |
| mema_sdram_parity_error1 | 1 | ROL | Parity error on the low byte of bank A SDRAM. |
| memb_sdram_parity_error0 | 2 | ROL | Parity error on the high byte of bank B SDRAM. |
| memb_sdram_parity_error1 | 3 | ROL | Parity error on the low byte of bank B SDRAM. |
| mema_ssram_parity_error0 | 4 | ROL | Parity error on the high byte of bank A SSRAM. |
| mema_ssram_parity_error1 | 5 | ROL | Parity error on the low byte of bank A SSRAM. |
| memb_ssram_parity_error0 | 6 | ROL | Parity error on the high byte of bank B SSRAM. |
| memb_ssram_parity_error1 | 7 | ROL | Parity error on the low byte of bank B SSRAM. |
| memb_bad_cpu_access | 8 | ROL | Indicates that an access was attempted to bank B when the bank was not present. |
| sdrama_too_late | 9 | ROL | Indicates that bank A SDRAM cannot refresh quickly enough. Information in that memory may be corrupt. |
| sdramb_too_late | 10 | ROL | Indicates that bank B SDRAM cannot refresh quickly enough. Information in that memory may be corrupt. |
| reserved | 15:11 | ROL | Reserved. Always read as “0000_0” |

**Table 45 - Main Status Register**

**Address: 204h**
**Label: status0_ie**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mema_sdram_parity_error0_ie | 0 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| mema_sdram_parity_error1_ie | 1 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| memb_sdram_parity_error0_ie | 2 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| memb_sdram_parity_error1_ie | 3 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| mema_ssram_parity_error0_ie | 4 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| mema_ssram_parity_error1_ie | 5 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| memb_ssram_parity_error0_ie | 6 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| memb_ssram_parity_error1_ie | 7 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| memb_bad_cpu_access_ie | 8 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| sdrama_too_late_ie | 9 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| sdramb_too_late_ie | 10 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| reserved | 15:11 | RO | Reserved. Always read as “0000_0” |

**Table 46 - Main Interrupt Enable Register**

**Address: 210h**
**Label: interrupt_flags**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cpureg_interrupt_active | 0 | RO | Indicates that the interrupt generated by this module is active. |
| mainreg_interrupt_active | 1 | RO | Indicates that the interrupt generated by this module is active. |
| txreg_interrupt_active | 2 | RO | Indicates that the interrupt generated by this module is active. |
| rxreg_interrupt_active | 3 | RO | Indicates that the interrupt generated by this module is active. |
| txtdmreg_interrupt_active | 4 | RO | Indicates that the interrupt generated by this module is active. |
| utoreg_interrupt_active | 5 | RO | Indicates that the interrupt generated by this module is active. |
| h100reg_interrupt_active | 6 | RO | Indicates that the interrupt generated by this module is active. |
| miscreg_interrupt_active | 7 | RO | Indicates that the interrupt generated by this module is active. |
| rxtdmreg_interrupt_active | 8 | RO | Indicates that the interrupt generated by this module is active. |
| aal0alarm_interrupt_active | 9 | RO | Set when RX AAL0 data cell FIFO is half full or a time-out occurs. Clear by writing ‘1’ to register 468h bit ‘0’ |
| clkrecovalarm_interrupt_active | 10 | RO | Set when the clock recovery buffer A or buffer B is half-full. Automatically cleared when both buffers are less than half full. |
| erroralarm_interrupt_active | 11 | RO | Set when the Error/Event FIFO is half full or a time-out occurs. Clear by writing ‘1’ to register 468h bit ‘1’ |
| reserved | 14:12 | RO | Reserved. Always Read as “000” |
| interrupt1_treated | 15 | PUL | When written to '1', another interrupt will not be generates for the number of us indicated in reg 214h. |

**Table 47 - Interrupt Flag Register**

**Address: 214h**
**Label: interrupt1_conf**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| min_interrupt1_period | 13:0 | RW | Number of us between interrupts (minimum). When 0000h, these is no minimum interval between interrupts. |
| interrupt1_polarity | 15:14 | RW | Interrupt polarity and output enable. “00”=active low (open-collector); “01”=active high (open-collector); “10” = drive low; “11” = drive high. |

**Table 48 - Minimum Interrupt Interval Register**

**Address: 216h**
**Label: interrupt2_conf**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| reserved | 13:0 | RW | Reserved. Must always be “0000_0000_0000_00” |
| interrupt2_polarity | 15:14 | RW | Interrupt polarity and output enable. “00”=active low (open-collector); “01”=active high (open-collector); “10” = drive low; “11” = drive high. |

**Table 49 - Interrupt Polarity & O/P Enable Register**

**Address: 218h**
**Label: interrupt1_enable**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cpureg_interrupt1_enable | 0 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |
| mainreg_interrupt1_enable | 1 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |
| txreg_interrupt1_enable | 2 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |
| rxreg_interrupt1_enable | 3 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |
| txtdmreg_interrupt1_enable | 4 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |
| utoreg_interrupt1_enable | 5 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |
| h100reg_interrupt1_enable | 6 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |
| miscreg_interrupt1_enable | 7 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |

**Table 50 - Interrupt 1 Enable Register**

| | | | |
|---|---|---|---|
| rxtdmreg_interrupt1_enable | 8 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |
| aal0alarm_interrupt1_enable | 9 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |
| clkrecovalarm_interrupt1_enable | 10 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |
| erroralarm_interrupt1_enable | 11 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt1 will be active. |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 50 - Interrupt 1 Enable Register (continued)**

**Address: 21Ah**
**Label: interrupt2_enable**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cpureg_interrupt2_enable | 0 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| mainreg_interrupt2_enable | 1 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| txreg_interrupt2_enable | 2 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| rxreg_interrupt2_enable | 3 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| txtdmreg_interrupt2_enable | 4 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| utoreg_interrupt2_enable | 5 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| h100reg_interrupt2_enable | 6 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| miscreg_interrupt2_enable | 7 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| rxtdmreg_interrupt2_enable | 8 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| aal0alarm_interrupt2_enable | 9 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| clkrecovalarm_interrupt2_enable | 10 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| erroralarm_interrupt2_enable | 11 | RW | When '1' and the corresponding active bit in reg 210h is active, interrupt2 will be active. |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 51 - Interrupt 2 Enable Register**

**Address: 220h**
**Label: txa_clk_gen**
**Reset Value: 0403h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| txa_clk_div[5:0] | 5:0 | RW | txa_clk clock source division value. The txa_clk clock source (selected using txa_clk_src) can be divided before being sent out on UTOPIA. Note that odd values will force the duty cycle to be maintained, rather than returning it to 50-50. |
| txa_clk_divisor_load_now | 6 | PUL | This bit, when written to '1', will force the new txa_clk_div to be applied immediately (possibly causing glitches on the txa_clk). This bit should only be set to one when loading the divisor when the txa_clk_present bit is cleared. |
| | | | Note that it is possible to dynamically change the divisor value without causing glitches on the output clock if this bit is not written to 1. |
| txa_clk_inv | 7 | RW | When '1', the txa_clk's source will be inverted before being driven out on the txa_clk pin. |
| txa_clk_src[2:0] | 10:8 | RW | “000” =txa_clk_in; “001” =txb_clk_in; ”010”=txc_clk_in; “011”=rxa_clk_in; “100” =rxb_clk_in; “101”=rxc_clk_in; “110”=mem_clk; others=reserved. |
| txa_clk_oe | 11 | RW | txa_clk output enable. Active high. |
| txa_clk_present | 12 | RW | Set to '1' when the txa_clk is present. If the user does not want to use the txa UTOPIA interface, this bit should be left at '0' regardless of the presence of the txa_clk. |
| txa_clk_divisor_reset | 13 | RW | When '0', will reset the clock division. |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 52 - Tx A Clock Division Register**

**Address: 222h**
**Label: txb_clk_gen**
**Reset Value: 0403h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| txb_clk_div[5:0] | 5:0 | RW | txb_clk clock source division value. The txb_clk clock source (selected using txb_clk_src) can be divided before being sent out on UTOPIA. Note that odd values will force the duty cycle to be maintained, rather than returning it to 50-50. |
| txb_clk_divisor_load_now | 6 | PUL | This bit, when written to '1', will force the new txb_clk_div to be applied immediately (possibly causing glitches on the txb_clk). This bit should only be set to one when loading the divisor when the txb_clk_present bit is cleared.<br>Note that it is possible to dynamically change the divisor value without causing glitches on the output clock if this bit is not written to 1. |
| txb_clk_inv | 7 | RW | When '1', the txb_clk's source will be inverted before being driven out on the txb_clk pin. |
| txb_clk_src[2:0] | 10:8 | RW | “000”=txa_clk_in; “001” =txb_clk_in; “010”=txc_clk_in; “011”=rxa_clk_in; “100”=rxb_clk_in; “101”=rxc_clk_in; “110”=mem_clk; others=reserved. |
| txb_clk_oe | 11 | RW | txb_clk output enable. Active high. |
| txb_clk_present | 12 | RW | Set to '1' when the txb_clk is present. If the user does not want to use the txa UTOPIA interface, this bit should be left at '0' regardless of the presence of the txb_clk. |
| txb_clk_divisor_reset | 13 | RW | When '0', will reset the clock division. |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 53 - Tx B Clock Division Register**

**Address: 224h**
**Label: txc_clk_gen**
**Reset Value: 0403h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| txc_clk_div[5:0] | 5:0 | RW | txc_clk clock source division value. The txc_clk clock source (selected using txc_clk_src) can be divided before being sent out on UTOPIA. Note that odd values will force the duty cycle to be maintained, rather than returning it to 50-50. |
| txc_clk_divisor_load_now | 6 | PUL | This bit, when written to '1', will force the new txc_clk_div to be applied immediately (possibly causing glitches on the txc_clk). This bit should only be set to one when loading the divisor when the txc_clk_present bit is cleared. |
| | | | Note that it is possible to dynamically change the divisor value without causing glitches on the output clock if this bit is not written to 1. |
| txc_clk_inv | 7 | RW | When '1', the txc_clk's source will be inverted before being driven out on the txc_clk pin. |
| txc_clk_src[2:0] | 10:8 | RW | “000”=txa_clk_in; “001”=txb_clk_in; ”010”=txc_clk_in; “011”=rxa_clk_in; “100”=rxb_clk_in; “101”=rxc_clk_in; “110”=mem_clk; others = reserved. |
| txc_clk_oe | 11 | RW | txc_clk output enable. Active high. |
| txc_clk_present | 12 | RW | Set to '1' when the txc_clk is present. If the user does not want to use the txa UTOPIA interface, this bit should be left at '0' regardless of the presence of the txc_clk. |
| txc_clk_divisor_reset | 13 | RW | When '0', will reset the clock division. |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 54 - Tx C Clock Division Register**

**Address: 228h**
**Label: rxa_clk_gen**
**Reset Value: 0403h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rxa_clk_div[5:0] | 5:0 | RW | rxa_clk clock source division value. The rxa_clk clock source (selected using rxa_clk_src) can be divided before being sent out on UTOPIA. Note that odd values will force the duty cycle to be maintained, rather than returning it to 50-50. |
| rxa_clk_divisor_load_now | 6 | PUL | This bit, when written to '1', will force the new rxa_clk_div to be applied immediately (possibly causing glitches on the rxa_clk). This bit should only be set to one when loading the divisor when the rxa_clk_present bit is cleared. |
| | | | Note that it is possible to dynamically change the divisor value without causing glitches on the output clock if this bit is not written to 1. |
| rxa_clk_inv | 7 | RW | When '1', the rxa_clk's source will be inverted before being driven out on the rxa_clk pin. |
| rxa_clk_src[2:0] | 10:8 | RW | “000”=txa_clk_in; “001”=txb_clk_in; “010”=txc_clk_in; “011”=rxa_clk_in; “100”=rxb_clk_in; “101”=rxc_clk_in; “110”=mem_clk; others=reserved. |
| rxa_clk_oe | 11 | RW | rxa_clk output enable. Active high. |
| rxa_clk_present | 12 | RW | Set to '1' when the rxa_clk is present. If the user does not want to use the txa UTOPIA interface, this bit should be left at '0' regardless of the presence of the rxa_clk. |
| rxa_clk_divisor_reset | 13 | RW | When '0', will reset the clock division. |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 55 - Rx A Clock Division Register**

**Address: 22Ah**
**Label: rxb_clk_gen**
**Reset Value: 0403h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rxb_clk_div[5:0] | 5:0 | RW | rxb_clk clock source division value. The rxb_clk clock source (selected using rxb_clk_src) can be divided before being sent out on UTOPIA. Note that odd values will force the duty cycle to be maintained, rather than returning it to 50-50. |
| rxb_clk_divisor_load_now | 6 | PUL | This bit, when written to '1', will force the new rxb_clk_div to be applied immediately (possibly causing glitches on the rxb_clk). This bit should only be set to one when loading the divisor when the rxb_clk_present bit is cleared. |
| | | | Note that it is possible to dynamically change the divisor value without causing glitches on the output clock if this bit is not written to 1. |
| rxb_clk_inv | 7 | RW | When '1', the rxb_clk's source will be inverted before being driven out on the rxb_clk pin. |
| rxb_clk_src[2:0] | 10:8 | RW | “000”=txa_clk_in; “001”=txb_clk_in; “010”=txc_clk_in; “011”=rxa_clk_in; “100”=rxb_clk_in; “101”=rxc_clk_in; “110”=mem_clk; others=reserved. |
| rxb_clk_oe | 11 | RW | rxb_clk output enable. Active high. |
| rxb_clk_present | 12 | RW | Set to '1' when the rxb_clk is present. If the user does not want to use the txa UTOPIA interface, this bit should be left at '0' regardless of the presence of the rxb_clk. |
| rxb_clk_divisor_reset | 13 | RW | When '0', will reset the clock division. |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 56 - Rx B Clock Division Register**

**Address: 22Ch**
**Label: rxc_clk_gen**
**Reset Value: 0403h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rxc_clk_div[5:0] | 5:0 | RW | rxc_clk clock source division value. The rxc_clk clock source (selected using rxc_clk_src) can be divided before being sent out on UTOPIA. Note that odd values will force the duty cycle to be maintained, rather than returning it to 50-50. |
| rxc_clk_divisor_load_now | 6 | PUL | This bit, when written to '1', will force the new rxc_clk_div to be applied immediately (possibly causing glitches on the rxc_clk). This bit should only be set to one when loading the divisor when the rxc_clk_present bit is cleared. |
| | | | Note that it is possible to dynamically change the divisor value without causing glitches on the output clock if this bit is not written to 1. |
| rxc_clk_inv | 7 | RW | When '1', the rxc_clk's source will be inverted before being driven out on the rxc_clk pin. |
| rxc_clk_src[2:0] | 10:8 | RW | “000”=txa_clk_in; “001”=txb_clk_in; “010”=txc_clk_in; “011”=rxa_clk_in; “100”=rxb_clk_in; “101”=rxc_clk_in; “110”=mem_clk; others = reserved. |
| rxc_clk_oe | 11 | RW | rxc_clk output enable. Active high. |
| rxc_clk_present | 12 | RW | Set to '1' when the rxc_clk is present. If the user does not want to use the txa UTOPIA interface, this bit should be left at '0' regardless of the presence of the rxc_clk. |
| rxc_clk_divisor_reset | 13 | RW | When '0', will reset the clock division. |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 57 - Rx C Clock Division Register**

**Address: 230h**
**Label: mem_parity0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mema_parity_conf[1:0] | 1:0 | RW | Selects how the parity bits of memory bank A are used (either as data bits or as parity bits). They should be set to “11” to allow correct Underrun detection in the RX TDM control memory. '0' = parity bits; '1' = user data. |
| memb_parity_conf[1:0] | 3:2 | RW | Selects how the parity bits of memory bank B are used (either as data bits or as parity bits). They should be set to “11” to allow correct Underrun detection in the RX TDM control memory. '0' = parity bits; '1' = user data. |
| cpu_parity_conf | 4 | RW | 1' = automatic parity calculation/checking on CPU accesses; '0' = CPU parity on CPU accesses (read and writes). |
| reserved | 7:5 | RO | Reserved. Always read as “00” |
| mem_parity_generation_data_mask | 15:8 | RW | The data bits whose corresponding bit is set to '1' in this vector will be used in parity calculation to the external memory. |

**Table 58 - Memory Parity Register 0**

**Address: 232h**
**Label: mem_parity1**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mem_parity_generation_add_mask[22:16] | 6:0 | RW | The address bits whose corresponding bit is set to '1' in this vector will be used in parity calculation to the external memory. |
| reserved | 15:7 | RW | Reserved. Must always be “0000_0000_0” |

**Table 59 - Memory Parity Register 1**

**Address: 234h**
**Label: mem_parity2**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mem_parity_generation_add_mask[15:0] | 15:0 | RW | The address bits whose corresponding bit is set to '1' in this vector will be used in parity calculation to the external memory. |

**Table 60 - Memory Parity Register 2**

**Address: 240h**
**Label: mem_conf0**
**Reset Value: 0075h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mema_add_lines | 1:0 | RW | “11” = 1 Mb per chip; “10” = 512 Kb per chip; “01” = 256Kb per chip; “00” = 128 Kb per chip |
| memb_add_lines | 3:2 | RW | “11” = 1 Mb per chip; “10” = 512 Kb per chip; “01” = 256Kb per chip; “00” = 128 Kb per chip |
| sar_capacity | 5:4 | RW | “00”=128 channels; “01”=256 channels; “10”=512 channels; “11”=1023 channel. |
| rx_circular_buffer_size | 7:6 | RW | Indicates the size of the RX Circular Buffers in external memory. They can be set locally to 256, 512, 1024 bytes. “00”= reserved;”01”=256 bytes; “10”=512 bytes; “11”=1024 bytes. |
| memb_bank_present | 8 | RW | Indicates if memory bank B is present or not. '0' = memb bank absent; '1' = memb bank present. |
| reserved | 15:9 | RW | Reserved. Must always be “0000_000” |

**Table 61 - Memory Configuration Register 0**

**Address: 242h**
**Label: mem_conf1**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rx_base_address | 7:0 | RW | Base address of the RX SSRAM in Bank A. Specified in increments of 8K bytes. Not used if bank B is present. |
| reserved | 15:8 | RW | Reserved. Must always be “0000_0000” |

**Table 62 - Memory Configuration Register 1**

**Address: 250h**
**Label: sdram_conf0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| sdrama_enable | 0 | RW | When '0', the values placed in the sdram_conf3 register will be placed on the SDRAM A pins. |
| sdramb_enable | 1 | RW | When '0', the values placed in the sdram_conf3 register will be placed on the SDRAM B pins. |
| sdrama_manual_access | 2 | PUL | When written to 1 and sdrama_enable is '0', the values placed in the sdram_conf4 register will be placed on the SDRAM A pins. |
| sdramb_manual_access | 3 | PUL | When written to 1 and sdramb_enable is '0', the values placed in the sdram_conf4 register will be placed on the SDRAM B pins. |
| sdrama_size | 4 | RW | 0' = 4M x 16 (8 Megabytes), '1' = 8M x 16 (16 Megabytes) |
| sdramb_size | 5 | RW | 0' = 4M x 16 (8 Megabytes), '1' = 8M x 16 (16 Megabytes) |
| sdram_refresh_freq | 7:6 | RW | “00” = 1 refresh every 16 cycles, “01” = 1 refresh every 8 cycles, “10” = 1 refresh every 4 cycles. Typical value “00”. |
| reserved | 15:8 | RW | Reserved. Must always be “0000_0000”. |

**Table 63 - SDRAM Configuration Register 0**

**Address: 252h**
**Label: sdram_conf1**
**Reset Value: 0400h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| sdram_refresh_cnt | 15:0 | RW | Number of mem_clk cycles per refresh to the SDRAM. |

**Table 64 - SDRAM Configuration Register 1**

**Address: 254h**
**Label: sdram_conf2**
**Reset Value: 0200h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| sdram_max_lateness | 15:0 | RW | Maximum number of refreshes that the SDRAM can be late before reporting an error. |

**Table 65 - SDRAM Configuration Register 2**

**Address: 256h**
**Label: sdram_conf3**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| sdram_add_idle | 13:0 | RW | While sdram_enable is '0', the value placed in this register will be put on the address lines. |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 66 - SDRAM Configuration Register 3**

**Address: 258h**
**Label: sdram_conf4**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| sdram_add_man | 13:0 | RW | While sdram_enable is '0', the value placed in this register will be put on the address lines for 1 cycle if sdram_manual_access is written to '1'. |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 67 - SDRAM Configuration Register 4**

**Address: 25Ah**
**Label: sdram_conf5**
**Reset Value: 003Fh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| sdram_cas_idle | 0 | RW | While sdram_enable is '0', the value placed in this register will be put on the cas line. |
| sdram_ras_idle | 1 | RW | While sdram_enable is '0', the value placed in this register will be put on the ras line. |
| sdram_we_idle | 2 | RW | While sdram_enable is '0', the value placed in this register will be put on the we line. |
| sdram_cas_man | 3 | RW | While sdram_enable is '0', the value placed in this register will be put on the cas line for 1 cycle if sdram_manual_access is written to '1'. |
| sdram_ras_man | 4 | RW | While sdram_enable is '0', the value placed in this register will be put on the ras line for 1 cycle if sdram_manual_access is written to '1'. |
| sdram_we_man | 5 | RW | While sdram_enable is '0', the value placed in this register will be put on the we line for 1 cycle if sdram_manual_access is written to '1'. |
| reserved | 15:6 | RW | Reserved. Must always be “0000_0000_00” |

**Table 68 - SDRAM Configuration Register 5**

### 3.3 TX Registers

**Address: 300h**
**Label: control**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| scanning_enable | 0 | RW | When '1', the cell time-out scanning process is enabled and will send cells beyond a certain pending time (programmed per VC). |
| aal0_cell_written | 1 | PUL | Written to '1' after an AAL0 cell has been written by the CPU to the TX AAL0 Cell FIFO. The cell will then be automatically sent. |
| reserved | 14:2 | RW | Reserved. Must always be “0000_0000_0000_0” |
| test_status | 15 | TS | Reserved. Must always be “0”. |

**Table 69 - TX Control Register**

**Address: 302h**
**Label: status0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| aal0_cell_fifo_empty | 0 | ROL | When '1', CPU can write another 4 AAL0 cells to internal FIFO |
| reserved | 15:1 | ROL | Reserved. Always read as “0000_0000_0000_000” |

**Table 70 - TX Status Register**

**Address: 304h**
**Label: status0_ie**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| aal0_cell_fifo_empty_ie | 0 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| reserved | 15:1 | RO | Reserved. Always read as “0000_0000_0000_000” |

**Table 71 - TX Interrupt Enable Register**

**Address: 310h**
**Label: aal0_monitor**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| aal0_cell_write_ok | 0 | RO | When '1', CPU can write another AAL0 cell to internal FIFO |
| reserved | 15:1 | RO | Reserved. Always read as “0000_0000_0000_000” |

**Table 72 - TX AAL0 Monitor Register**

### 3.4 RX Registers

**Address: 400h**
**Label: control**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| hdlc_type | 0 | RW | ‘0'=Bit wide HDLC Framing; '1'=Byte Wide HDLC Framing. |
| packaging_type | 1 | RW | ‘0' = no AAL2 Header in HDLC Data Field; '1' = Raw AAL2 CPS-Packet in HDLC Data Field |
| nibble_mode | 2 | RW | ‘0' = High bits used for ADPCM nibble, '1' = low bits used for ADPCM nibble |
| reserved | 14:3 | RW | Reserved. Must Always be “0000_0000_0000” |
| test_status | 15 | TS | Reserved. Must always be “0”. |

**Table 73 - RX Control Register**

**Address: 402h**
**Label: status0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cpu_buffer_overflow | 0 | ROL | CPU Destined CPS-Packet Buffer Overflow. |
| aal0_overflow | 1 | ROL | Overflow in the receive AAL0 cell buffer. |
| error_overflow | 2 | ROL | Overflow in the error/event structure buffer. |
| reserved | 15:3 | ROL | Reserved. Always read as “0000_0000_0000_0” |

**Table 74 - RX Status Register**

**Address: 404h**
**Label: status0_ie**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cpu_buffer_overflow_ie | 0 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| aal0_overflow_ie | 1 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| error_overflow_ie | 2 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| reserved | 15:3 | RO | Reserved. Always read as “0000_0000_0000_0” |

**Table 75 - RX Interrupt Enable Register**

**Address: 410h**
**Label: silent_pattern_reg**
**Reset Value: 00FFh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| write_back_pattern | 7:0 | RW | PCM Silent pattern written back to RX circular buffers by RX TDM module (only if enabled in PCM RX Control Memory structure) |
| reserved | 15:8 | RW | Reserved. Must always be “0000_0000” |

**Table 76 - PCM Silent Pattern Register**

**Address: 428h**
**Label: crc_config0**
**Reset Value: FFFFh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| crc_preset | 15:0 | RW | HDLC CRC Preset. Must be “1111_1111_1111_1111” for ITU compliance. |

**Table 77 - CRC Configuration Register 0**

**Address: 42Ah**
**Label: crc_config1**
**Reset Value: F0B8h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| crc_mask | 15:0 | RW | HDLC CRC End Mask. Must be “1111_0000_1011_1000” for ITU compliance. |

**Table 78 - CRC Configuration Register 1**

**Address: 430h**
**Label: aal0_manage**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rx_aal0_base_add | 8:0 | RW | Represents bits 20:12 of the byte address at which the AAL0 data cell FIFO is mapped. |
| rx_aal0_buf_size | 11:9 | RW | “000” = 4KByte, “001” = 8KB, “010” = 16KB, “011” = 32KB, “100” = 64KB, “101” = 128KB, others = reserved |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 79 - AAL0 FIFO Management Register**

**Address: 432h**
**Label: aal0_read**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rx_aal0_rpnt | 11:0 | RW | The CPU's read pointer to the AAL0 data cell FIFO. This read pointer has an extra bit to allow for a full buffer (for example, 128 cells instead of 127) |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 80 - AAL0 Read Pointer Register**

**Address: 434h**
**Label: aal0_write**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rx_aal0_wpnt | 11:0 | RO | The hardware's write pointer to the AAL0 data cell FIFO. This write pointer has an extra bit to allow for a full buffer (for example, 128 cells instead of 127) |
| reserved | 15:12 | RO | Reserved. Always read as “0000” |

**Table 81 - AAL0 Write Pointer Register**

**Address: 438h**
**Label: error_manage**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| error_base_add | 8:0 | RW | Represents bits 20:12 of the byte address at which the error structure buffer is mapped. |
| error_buf_size | 11:9 | RW | “000” = 4KByte, “001” = 8KB, “010” = 16KB, “011” = 32KB, “100” = 64KB, “101” = 128KB, others = reserved |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 82 - Error Management Register**

**Address: 43Ah**
**Label: error_write**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| error_wpnt | 14:0 | RO | The chip's write pointer to the error structure buffer. This write pointer has an extra bit to allow for a full buffer (for example, 1024 structures instead of 1023) |
| reserved | 15 | RO | Reserved. Always read as “0” |

**Table 83 - Error Write Pointer Register**

**Address: 43Ch**
**Label: error_readure**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| error_rpnt | 14:0 | RW | The CPU's read pointer to the error structure buffer. This read pointer has an extra bit to allow for a full buffer (for example, 1024 structures instead of 1023) |
| reserved | 15 | RW | Reserved. Must always be “0” |

**Table 84 - Error Read Pointer Register**

**Address: 440h**
**Label: cpu_manage0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cpu_cpsp_base_add | 6:0 | RW | Bits 20:14 of the byte address at which the CPU-destined CPS-Packet buffer is mapped. |
| cpu_cpsp_buf_size | 8:7 | RW | “00” = 16KByte, “01” = 32KB, “10” = 64KB, “11” = 128KB |
| reserved | 15:9 | RW | Reserved. Must always be “0000_000” |

**Table 85 - CPU Management Register 0**

**Address: 442h**
**Label: cpu_manage1**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cpu_cpsp_rpnt | 15:0 | RW | Word read pointer to the CPU-destined CPS-Packet buffer |

**Table 86 - CPU Management Register 1**

**Address: 460h**
**Label: aal0_timeout_period_high**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| aal0_timeout_period[19:16] | 3:0 | RW | If an AAL0 cell remains in the buffer for the period in this register or more, an alarm will be generated. In *u*s. |
| reserved | 15:4 | RW | Reserved. Must always be “0000_0000_0000” |

**Table 87 - AAL0 Timeout Period (High Word) Register**

**Address: 462h**
**Label: aal0_timeout_period_low**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| aal0_timeout_period[15:0] | 15:0 | RW | If an AAL0 cell remains in the buffer for the period in this register or more, an alarm will be generated. In *u*s. |

**Table 88 - AAL0 Timeout Period (Low Word) Register**

**Address: 464h**
**Label: error_timeout_period_high**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| error_timeout_period[19:16] | 3:0 | RW | This timeout period applies both to error reporting structures as well as to CPU-destined CPS-Packets. In *u*s. |
| reserved | 15:4 | RW | Reserved. Must always be “0000_0000_0000” |

**Table 89 - Error Timeout Period (High Word) Register**

**Address: 466h**
**Label: error_timeout_period_low**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| error_timeout_period[15:0] | 15:0 | RW | This timeout period applies both to error reporting structures as well as to CPU-destined CPS-Packets. In *u*s. |

**Table 90 - Error Timeout Period (Low Word) Register**

**Address: 468h**
**Label: pulse_register**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| aal0_treated_pulse | 0 | PUL | When the AAL0 buffer has been treated, write this bit to '1' to clear the alarm. |
| error_treated_pulse | 1 | PUL | When the error buffer has been treated, write this bit to '1' to clear the alarm. |
| reserved | 15:2 | PUL | Reserved. Always read as “0000_0000_0000_00” |

**Table 91 - AAL0 & Error Treated Register**

### 3.5 TX TDM Registers

**Address: 500h**
**Label: control**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| nibble_mode | 0 | RW | 0' = High bits used for ADPCM nibble, '1' = low bits used for ADPCM nibble |
| hdlc_type | 1 | RW | HDLC framing method. '0'=Bit wise HDLC Framing; '1'=Byte Wise HDLC Framing. |
| reserved | 2 | RW | Reserved. Must be “0”. |
| packaging_type | 3 | RW | 0' = AAL2 header not present in HDLC data field; '1' = AAL2 header present in HDLC data field. |
| llman_process_enable | 4 | RW | Enables the treatment of incoming TDM bytes from the H100 bus. |
| silent_bit_position | 7:5 | RW | Indicates the position of the simple silence bit in the associated odd stream. “000” = bit 7, “111” = bit 0 |
| u_law_zero_illegal | 8 | RW | When '1', 00h in u-law will be replaced by 02h before it is placed on RX TDM bus. |
| reserved | 14:9 | RW | Reserved. Must always be “0000_0000_00” |
| test_status | 15 | TS | Reserved. Must always be “0”. |

**Table 92 - TX TDM Control Register**

**Address: 502h**
**Label: status0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| llman_process_crashed | 0 | ROL | Indicates that the linked-list manager crashed. This indicates corrupt data in the linked list memory. |
| copying_out_of_bandwidth | 1 | ROL | Data copying process between the H.100 interface and the TX TDM was not fast enough. Indicates too low an mem_clk speed for the amount of TDM bandwidth. |
| frame_reading_out_of_bandwidth | 2 | ROL | Means that the frame reading process that reads from the frame memory fell out of sync with the TDM bus. This error will occur naturally at startup. |
| bad_packet_length | 3 | ROL | Received packet length on HDLC exceeded 64 bytes. |
| circular_buffer_overflow | 4 | ROL | TX SAR did not read data fast enough from circular buffer. Bad data integrity will ensue. |
| cps_packet_refused_error | 5 | ROL | CPS-Packet could not be written because all 512 descriptors for the VC were occupied. |
| txsar_wbcache_overflow | 6 | ROL | The 64 X 21 TX SAR write back cache overflow. |

**Table 93 - TDM TX Status Register**

| | | | |
|---|---|---|---|
| txtdm_wcache_overflow | 7 | ROL | The 128 X 72 byte write cache overflow. |
| packet_fifo_overflow | 8 | ROL | The 1024 X 41 packet FIFO overflow. |
| misaligned_flag | 9 | ROL | HDLC Packet got a misaligned flag (not aligned on a byte boundary) |
| bad_idle_code | 10 | ROL | HDLC Packet got idle code in the middle of a packet! |
| short_packet | 11 | ROL | HDLC Packet was too short (0 data bytes!) |
| dcoffset_overflow | 12 | ROL | DC offset value caused a linear value to be calculated as more than 4096 |
| Reserved | 15:13 | ROL | Reserved. Always read as “000” |

**Table 93 - TDM TX Status Register (continued)**

**Address: 504h**
**Label: status0_ie**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| llman_process_crashed_ie | 0 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| copying_out_of_bandwidth_ie | 1 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| frame_reading_out_of_bandwidth_ie | 2 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| bad_packet_length_ie | 3 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| circular_buffer_overflow_ie | 4 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| cps_packet_refused_error_ie | 5 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| txsar_wbcache_overflow_ie | 6 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| txtdm_wcache_overflow_ie | 7 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| packet_fifo_overflow_ie | 8 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| misaligned_flag_ie | 9 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| bad_idle_code_ie | 10 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| short_packet_ie | 11 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| dcoffset_overflow_ie | 12 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| Reserved | 15:13 | RO | Reserved. Always read as “000” |

**Table 94 - TDM TX Interrupt Enable Register**

**Address: 512h**
**Label: silent_pattern_a**
**Reset Value: FFFFh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| silent_bytea_match | 7:0 | RW | Silent pattern detection match register A. When a received data byte matches the match and mask registers, it is tagged as silent. |
| silent_bytea_mask | 15:8 | RW | Silent pattern detection mask register A. When a received data byte matches the match and mask registers, it is tagged as silent. |

**Table 95 - Silent Pattern Detection Match Register A**

**Address: 514h**
**Label: silent_pattern_b**
**Reset Value: FFFFh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| silent_byteb_match | 7:0 | RW | Silent pattern detection match register B. When a received data byte matches the match and mask registers, it is tagged as silent. |
| silent_byteb_mask | 15:8 | RW | Silent pattern detection mask register B. When a received data byte matches the match and mask registers, it is tagged as silent. |

**Table 96 - Silent Pattern Detection Match Register B**

**Address: 518h**
**Label: tdm_pointer_monitor**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| global_tdm_pointer | 15:0 | RO | Current value of the global TDM pointer. |

**Table 97 - TDM Pointer Monitor Register**

**Address: 520h**
**Label: cpu_cps_packet0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cps_packet_cpu_vc_number | 9:0 | RW | VC Number on which a CPU sourced CPS-Packet must be sent. |
| cps_packet_cpu_request | 10 | PC | Set to '1' by the CPU when a CPU Sourced CPS-Packet is pending transmission. Clear by hardware, when the CPS-Packet has been queue for transmission. |
| reserved | 15:11 | PC | Reserved. Always read as “0000_0” |

**Table 98 - CPU CPS-Packet Register 0**

**Address: 522h**
**Label: cpu_cps_packet1**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cps_packet_cpu_descriptor[47:32] | 15:0 | RW | CPU Sourced CPS-Packet Descriptor. |

**Table 99 - CPU CPS-Packet Register 1**

**Address: 524h**
**Label: cpu_cps_packet2**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cps_packet_cpu_descriptor[31:16] | 15:0 | RW | CPU Sourced CPS-Packet Descriptor. |

**Table 100 - CPU CPS-Packet Register 2**

**Address: 526h**
**Label: cpu_cps_packet3**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cps_packet_cpu_descriptor[15:0] | 15:0 | RW | CPU Sourced CPS-Packet Descriptor. |

**Table 101 - CPU CPS-Packet Register 3**

### 3.6 UTOPIA Registers

**Address: 600h**
**Label: control**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rxa_ena | 0 | RW | 0' = RXA Disabled; '1' = RXA Operates Normally. |
| rxb_ena | 1 | RW | 0' = RXB Disabled; '1' = RXB Operates Normally. |
| rxc_ena | 2 | RW | 0' = RXC Disabled; '1' = RXC Operates Normally. |
| porta_sar_mode | 3 | RW | Puts TXA/RXA in PHY or SAR mode. '1'=PHY Mode; '0'=SAR Mode. |
| portb_sar_mode | 4 | RW | Puts TXB/RXB in PHY or SAR mode. '1'=PHY Mode; '0'=SAR Mode. |
| portc_sar_mode | 5 | RW | Puts TXC/RXC in PHY or SAR mode. '1'=PHY Mode; '0'=SAR Mode. |
| porta_level2_mode | 6 | RW | Puts TXA/RXA in UTOPIA Level 2 mode. This disables port B and requires port A to be configured as PHY. '0' = Level 1; '1' = Level 2. |
| txa_always_drive_dat_soc_par | 7 | RW | 0' = only drive when selected; '1' = always drive DAT/PAR/SOC pins. |
| txb_always_drive_dat_soc_par | 8 | RW | 0' = only drive when selected; '1' = always drive DAT/PAR/SOC pins. |
| txc_always_drive_dat_soc_par | 9 | RW | 0' = only drive when selected; '1' = always drive DAT/PAR/SOC pins. |
| clav_enb_oe | 10 | RW | When '0', tx_clav/tx_enb pins are in tri-state. |
| reserved | 14:11 | RW | Reserved. Must always be “0000” |
| test_status | 15 | TS | Reserved. Must always be “0”. |

**Table 102 - UTOPIA Control Register 1**

**Address: 602h**
**Label: status0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| phy_alarma | 0 | ROL | PHY alarm detected on port A |
| phy_alarmb | 1 | ROL | PHY alarm detected on port B |
| rx_cell_loss | 2 | ROL | Cell loss in RX SAR output FIFO |
| outa_cell_loss | 3 | ROL | Cell loss in port A output FIFO |
| outb_cell_loss | 4 | ROL | Cell loss in port B output FIFO |
| outc_cell_loss | 5 | ROL | Cell loss in port C output FIFO |
| cell_loss_rollover | 6 | CRL | Cell loss counter has wrapped |
| tx_arr_rollover | 7 | CRL | Counter of cells received from TX SAR has wrapped |
| rx_dep_rollover | 8 | CRL | Counter of cells sent to RX SAR has wrapped |
| ia_arr_rollover | 9 | CRL | Counter of cells received from port A has wrapped |
| oa_dep_rollover | 10 | CRL | Counter of cells sent to port A has wrapped |
| ib_arr_rollover | 11 | CRL | Counter of cells received from port B has wrapped |
| ob_dep_rollover | 12 | CRL | Counter of cells sent to port B has wrapped |
| ic_arr_rollover | 13 | CRL | Counter of cells received from port C has wrapped |
| oc_dep_rollover | 14 | CRL | Counter of cells sent to port C has wrapped |
| aal0_arr_rollover | 15 | CRL | Counter of cells received from AAL0 input FIFO has wrapped |

**Table 103 - UTOPIA Status Register 0**

**Address: 604h**
**Label: status0_ie**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| phy_alarma_ie | 0 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| phy_alarmb_ie | 1 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| rx_cell_loss_ie | 2 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| outa_cell_loss_ie | 3 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| outb_cell_loss_ie | 4 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| outc_cell_loss_ie | 5 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| cell_loss_rollover_ie | 6 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| tx_arr_rollover_ie | 7 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| rx_dep_rollover_ie | 8 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ia_arr_rollover_ie | 9 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| oa_dep_rollover_ie | 10 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ib_arr_rollover_ie | 11 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ob_dep_rollover_ie | 12 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ic_arr_rollover_ie | 13 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| oc_dep_rollover_ie | 14 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| aal0_arr_rollover_ie | 15 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |

**Table 104 - UTOPIA Interrupt Enable Register 0**

**Address: 60Ah**
**Label: status2**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| reserved | 2:0 | ROL | Reserved. Always read as “000” |
| rxa_parity_error | 3 | ROL | Parity error on UTOPIA port A |
| rxb_parity_error | 4 | ROL | Parity error on UTOPIA port B |
| rxc_parity_error | 5 | ROL | Parity error on UTOPIA port C |
| reserved | 15:6 | ROL | Reserved. Always read as “0000_0000_00” |

**Table 105 - UTOPIA Status Register 2**

**Address: 60Ch**
**Label: status2_ie**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| reserved | 2:0 | RO | Reserved. Always read as “000” |
| rxa_parity_error_ie | 3 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| rxb_parity_error_ie | 4 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| rxc_parity_error_ie | 5 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| reserved | 15:6 | RO | Reserved. Always read as “0000_0000_00” |

**Table 106 - UTOPIA Interrupt Enable Register 2**

**Address: 60Eh**
**Label: control2**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rxa_null_elim | 1:0 | RW | Port A Null Cell Elimination. “00”=no Null-Cell Deletion; “01” = reserved; “10” = Delete Null-Cells (UNI); “11” = Delete Null-Cells (NNI). |
| rxb_null_elim | 3:2 | RW | Port B Null Cell Elimination. “00”=no Null-Cell Deletion; “01” = reserved; “10” = Delete Null-Cells (UNI); “11” = Delete Null-Cells (NNI). |
| rxc_null_elim | 5:4 | RW | Port C Null Cell Elimination. “00”=no Null-Cell Deletion; “01” = reserved; “10” = Delete Null-Cells (UNI); “11” = Delete Null-Cells (NNI). |
| phy_alarm_mode | 7:6 | RW | “00” = Never detect PHY Alarm; “01” = active-high signal; “10” = active-low signal; “11” = reserved. |
| reserved | 15:8 | RW | Reserved. Must always be “0000_0000” |

**Table 107 - UTOPIA Control Register 2**

**Address: 612h**
**Label: cell_loss_counter**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cell_loss | 15:0 | CNT | Counter of the number of cells lost in all UTOPIA output FIFOs |

**Table 108 - Lost Cells Counter**

**Address: 614h**
**Label: gpio0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| phya_tx_led_conf | 1:0 | RW | “00”=Tri-state; “01” = used as LED; “10”=Drive Low; “11”=Drive high. |
| phya_rx_led_conf | 3:2 | RW | “00”=Tri-state; “01” = used as LED; “10”=Drive Low; “11”=Drive high. |
| phyb_tx_led_conf | 5:4 | RW | “00”=Tri-state; “01” = used as LED; “10”=Drive Low; “11”=Drive high. |
| phyb_rx_led_conf | 7:6 | RW | “00”=Tri-state; “01” = used as LED; “10”=Drive Low; “11”=Drive high. |
| phya_alm_input | 8 | RO | Indicates if the PHY alarm on port A is currently active |
| phyb_alm_input | 9 | RO | Indicates if the PHY alarm on port B is currently active |
| phya_tx_led_input | 10 | RO | Indicates if the TX LED on port A is currently active |
| phya_rx_led_input | 11 | RO | Indicates if the RX LED on port A is currently active |
| phyb_tx_led_input | 12 | RO | Indicates if the TX LED on port B is currently active |
| phyb_rx_led_input | 13 | RO | Indicates if the RX LED on port B is currently active |
| reserved | 15:14 | RO | Reserved. Always read as “00” |

**Table 109 - General Purpose I/O Register 0**

**Address: 620h**
**Label: porta_look_up_base**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| luta_base | 3:0 | RW | LUT Base address in megabytes. |
| luta_size | 5:4 | RW | “00”=128K VC; “01”=256K VC; “10”=512K VC; “11”=1024K VC. |
| reserved | 15:6 | RW | Reserved. Must always be “0000_0000_00” |

**Table 110 - Port A LUT Base Address Register**

**Address: 622h**
**Label: porta_concatenation**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vci_na | 4:0 | RW | Number of VCI bits in look-up table. |
| reserved | 15:5 | RW | Reserved. Must always be “0000_0000_000” |

**Table 111 - Port A VCI Bits in LUT**

**Address: 624h**
**Label: porta_vpi_match**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vpi_matcha | 11:0 | RW | For a cell from port A to be considered valid, any bits in its VPI whose corresponding bits in reg 626h are '1' must have the value contained in this register. |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 112 - Port A VPI Match Register**

**Address: 626h**
**Label: porta_vpi_mask**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vpi_maska | 11:0 | RW | For a cell from port A to be considered valid, any bits in its VPI whose corresponding bits in this register are '1' must have the value contained in reg 624h. |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 113 - Port A VPI Mask Register**

**Address: 628h**
**Label: porta_vci_match**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vci_matcha | 15:0 | RW | For a cell from port A to be considered valid, any bits in its VCI whose corresponding bits in reg 62Ah are '1' must have the value contained in this register. |

**Table 114 - Port A VCI Match Register**

**Address: 62Ah**
**Label: porta_vci_mask**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vci_maska | 15:0 | RW | For a cell from port A to be considered valid, any bits in its VCI whose corresponding bits in this register are '1' must have the value contained in reg 628h. |

**Table 115 - Port A VCI Mask Register**

**Address: 62Ch**
**Label: porta_overflow0**
**Reset Value: FFFFh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ia_rx_cell_max | 5:0 | RW | If the cell fill of the RX SAR output FIFO becomes greater than this value, cells from the port A input FIFO will be blocked. 3Fh = no backpressure |
| ia_oa_cell_max | 9:6 | RW | If the cell fill of the port A output FIFO becomes greater than this value, cells from the port A input FIFO will be blocked. Fh = no backpressure |
| ia_ob_cell_max | 13:10 | RW | If the cell fill of the port B output FIFO becomes greater than this value, cells from the port A input FIFO will be blocked. Fh = no backpressure |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 116 - Port A Overflow Register 0**

**Address: 62Eh**
**Label: porta_overflow1**
**Reset Value: 000Fh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ia_oc_cell_max | 3:0 | RW | If the cell fill of the port C output FIFO becomes greater than this value, cells from the port A input FIFO will be blocked. Fh = no backpressure |
| reserved | 15:4 | RW | Reserved. Must always be “0000_0000_0000” |

**Table 117 - Port A Overflow Register 1**

**Address: 630h**
**Label: porta_cell_arrival_high**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ia_arr[31:16] | 15:0 | CNT | Counter of the number of cells received from port A |

**Table 118 - Port A Cell Arrival Counter (High Word)**

**Address: 632h**
**Label: porta_cell_arrival_low**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ia_arr[15:0] | 15:0 | CNT | Counter of the number of cells received from port A |

**Table 119 - Port A Cell Arrival Counter (Low Word)**

**Address: 634h**
**Label: porta_cell_departure_high**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| oa_dep[31:16] | 15:0 | CNT | Counter of the number of cells sent onto port A |

**Table 120 - Port A Cell Departure Counter (High Word)**

**Address: 636h**
**Label: porta_cell_departure_low**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| oa_dep[15:0] | 15:0 | CNT | Counter of the number of cells sent onto port A |

**Table 121 - Port A Cell Departure Counter (Low Word)**

**Address: 640h**
**Label: portb_look_up_base**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| lutb_base | 3:0 | RW | LUT Base address in megabytes. |
| lutb_size | 5:4 | RW | “00”=128K VC; “01”=256K VC; “10”=512K VC; “11”=1024K VC. |
| reserved | 15:6 | RW | Reserved. Must always be “0000_0000_00” |

**Table 122 - Port B LUT Base Address Register**

**Address: 642h**
**Label: portb_concatenation**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vci_nb | 4:0 | RW | Number of VCI bits in look-up table. |
| reserved | 15:5 | RW | Reserved. Must always be “0000_0000_000” |

**Table 123 - Port B VCI Bits in LUT**

**Address: 644h**
**Label: portb_vpi_match**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vpi_matchb | 11:0 | RW | For a cell from port B to be considered valid, any bits in its VPI whose corresponding bits in reg 646h are '1' must have the value contained in this register. |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 124 - Port B VPI Match Register**

**Address: 646h**
**Label: portb_vpi_mask**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vpi_maskb | 11:0 | RW | For a cell from port B to be considered valid, any bits in its VPI whose corresponding bits in this register are '1' must have the value contained in reg 644h. |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 125 - Port B VPI Mask Register**

**Address: 648h**
**Label: portb_vci_match**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vci_matchb | 15:0 | RW | For a cell from port B to be considered valid, any bits in its VCI whose corresponding bits in reg 64Ah are '1' must have the value contained in this register. |

**Table 126 - Port B VCI Match Register**

**Address: 64Ah**
**Label: portb_vci_mask**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vci_maskb | 15:0 | RW | For a cell from port B to be considered valid, any bits in its VCI whose corresponding bits in this register are '1' must have the value contained in reg 648h. |

**Table 127 - Port B VCI Mask Register**

**Address: 64Ch**
**Label: portb_overflow0**
**Reset Value: 1108h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ib_rx_cell_max | 5:0 | RW | If the cell fill of the RX SAR output FIFO becomes greater than this value, cells from the port B input FIFO will be blocked. 3Fh = no backpressure |
| ib_oa_cell_max | 9:6 | RW | If the cell fill of the port A output FIFO becomes greater than this value, cells from the port B input FIFO will be blocked. Fh = no backpressure |
| ib_ob_cell_max | 13:10 | RW | If the cell fill of the port B output FIFO becomes greater than this value, cells from the port B input FIFO will be blocked. Fh = no backpressure |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 128 - Port B Overflow Register 0**

**Address: 64Eh**
**Label: portb_overflow1**
**Reset Value: 004h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ib_oc_cell_max | 3:0 | RW | If the cell fill of the port C output FIFO becomes greater than this value, cells from the port B input FIFO will be blocked. Fh = no backpressure |
| reserved | 15:4 | RW | Reserved. Must always be “0000_0000_0000” |

**Table 129 - Port B Overflow Register 1**

**Address: 650h**
**Label: portb_cell_arrival_high**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ib_arr[31:16] | 15:0 | CNT | Counter of the number of cells received from port B |

**Table 130 - Port B Cell Arrival Counter (High Word)**

**Address: 652h**
**Label: portb_cell_arrival_low**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ib_arr[15:0] | 15:0 | CNT | Counter of the number of cells received from port B |

**Table 131 - Port B Cell Arrival Counter (Low Word)**

**Address: 654h**
**Label: portb_cell_departure_high**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ob_dep[31:16] | 15:0 | CNT | Counter of the number of cells sent onto port B |

**Table 132 - Port B Cell Departure Counter (High Word)**

**Address: 656h**
**Label: portb_cell_departure_low**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ob_dep[15:0] | 15:0 | CNT | Counter of the number of cells sent onto port B |

**Table 133 - Port B Cell Departure Counter (Low Word)**

**Address: 660h**
**Label: portc_look_up_base**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| lutc_base | 3:0 | RW | LUT Base address in megabytes. |
| lutc_size | 5:4 | RW | “00”=128K VC; “01”=256K VC; “10”=512K VC; “11”=1024K VC. |
| reserved | 15:6 | RW | Reserved. Must always be “0000_0000_00” |

**Table 134 - Port C LUT Base Address Register**

**Address: 662h**
**Label: portc_concatenation**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vci_nc | 4:0 | RW | Number of VCI bits in look-up table. |
| reserved | 15:5 | RW | Reserved. Must always be “0000_0000_000” |

**Table 135 - Port C VCI Bits in LUT**

**Address: 664h**
**Label: portc_vpi_match**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vpi_matchc | 11:0 | RW | For a cell from port C to be considered valid, any bits in its VPI whose corresponding bits in reg 666h are '1' must have the value contained in this register. |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 136 - Port C VPI Match Register**

**Address: 666h**
**Label: portc_vpi_mask**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vpi_maskc | 11:0 | RW | For a cell from port C to be considered valid, any bits in its VPI whose corresponding bits in this register are '1' must have the value contained in reg 664h. |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 137 - Port C VPI Mask Register**

**Address: 668h**
**Label: portc_vci_match**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vci_matchc | 15:0 | RW | For a cell from port C to be considered valid, any bits in its VCI whose corresponding bits in reg 66Ah are '1' must have the value contained in this register. |

**Table 138 - Port C VCI Match Register**

**Address: 66Ah**
**Label: portc_vci_mask**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| vci_maskc | 15:0 | RW | For a cell from port C to be considered valid, any bits in its VCI whose corresponding bits in this register are '1' must have the value contained in reg 668h. |

**Table 139 - Port C VCI Mask Register**

**Address: 66Ch**
**Label: portc_overflow0**
**Reset Value: 1108h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ic_rx_cell_max | 5:0 | RW | If the cell fill of the RX SAR output FIFO becomes greater than this value, cells from the port C input FIFO will be blocked. 3Fh = no backpressure |
| ic_oa_cell_max | 9:6 | RW | If the cell fill of the port A output FIFO becomes greater than this value, cells from the port C input FIFO will be blocked. Fh = no backpressure |
| ic_ob_cell_max | 13:10 | RW | If the cell fill of the port B output FIFO becomes greater than this value, cells from the port C input FIFO will be blocked. Fh = no backpressure |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 140 - Port C Overflow Register 0**

**Address: 66Eh**
**Label: portc_overflow1**
**Reset Value: 004h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ic_oc_cell_max | 3:0 | RW | If the cell fill of the port C output FIFO becomes greater than this value, cells from the port C input FIFO will be blocked. Fh = no backpressure |
| reserved | 15:4 | RW | Reserved. Must always be “0000_0000_0000” |

**Table 141 - Port C Overflow Register 1**

**Address: 670h**
**Label: portc_cell_arrival_high**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ic_arr[31:16] | 15:0 | CNT | Counter of the number of cells received from port C |

**Table 142 - Port C Cell Arrival Counter (High Word)**

**Address: 672h**
**Label: portc_cell_arrival_low**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ic_arr[15:0] | 15:0 | CNT | Counter of the number of cells received from port C |

**Table 143 - Port C Cell Arrival Counter (Low Word)**

**Address: 674h**
**Label: portc_cell_departure_high**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| oc_dep[31:16] | 15:0 | CNT | Counter of the number of cells sent onto port C |

**Table 144 - Port C Cell Departure Counter (High Word)**

**Address: 676h**
**Label: portc_cell_departure_low**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| oc_dep[15:0] | 15:0 | CNT | Counter of the number of cells sent onto port C |

**Table 145 - Port C Cell Departure Counter (Low Word)**

**Address: 680h**
**Label: aal0_cell_arrival_high**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| aal0_arr[31:16] | 15:0 | CNT | Counter of the number of cells received from AAL0 FIFO |

**Table 146 - AAL0 Cell Arrival Counter (High Word)**

**Address: 682h**
**Label: aal0_cell_arrival_low**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| aal0_arr[15:0] | 15:0 | CNT | Counter of the number of cells received from AAL0 FIFO |

**Table 147 - AAL0 Cell Arrival Counter (Low Word)**

**Address: 688h**
**Label: aal0_overflow0**
**Reset Value: 1108h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| aal0_rx_cell_max | 5:0 | RW | If the cell fill of the RX SAR output FIFO becomes greater than this value, cells from the AAL0 input FIFO will be blocked. 3Fh = no backpressure |
| aal0_oa_cell_max | 9:6 | RW | If the cell fill of the port A output FIFO becomes greater than this value, cells from the AAL0 input FIFO will be blocked. Fh = no backpressure |
| aal0_ob_cell_max | 13:10 | RW | If the cell fill of the port B output FIFO becomes greater than this value, cells from the AAL0 input FIFO will be blocked. Fh = no backpressure |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 148 - AAL0 Overflow Register 0**

**Address: 68Ah**
**Label: aal0_overflow1**
**Reset Value: 004h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| aal0_oc_cell_max | 3:0 | RW | If the cell fill of the port C output FIFO becomes greater than this value, cells from the AAL0 input FIFO will be blocked. Fh = no backpressure |
| reserved | 15:4 | RW | Reserved. Must always be “0000_0000_0000” |

**Table 149 - AAL0 Overflow Register 1**

**Address: 690h**
**Label: tx_sar_cell_arrival_high**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| tx_arr[31:16] | 15:0 | CNT | Counter of the number of cells received from TX SAR |

**Table 150 - TX_SAR Cell Arrival Counter (High Word)**

**Address: 692h**
**Label: tx_sar_cell_arrival_low**
**Reset Value: 0001h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| tx_arr[15:0] | 15:0 | CNT | Counter of the number of cells received from TX SAR. It is 1 after reset. |

**Table 151 - TX_SAR Cell Arrival Counter (Low Word)**

**Address: 694h**
**Label: rx_sar_cell_departure_high**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rx_dep[31:16] | 15:0 | CNT | Counter of the number of cells sent to RX SAR |

**Table 152 - RX_SAR Cell Departure Counter (High Word)**

**Address: 696h**
**Label: rx_sar_cell_departure_low**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rx_dep[15:0] | 15:0 | CNT | Counter of the number of cells sent to RX SAR |

**Table 153 - RX_SAR Cell Departure Counter (Low Word)**

**Address: 698h**
**Label: tx_sar_overflow0**
**Reset Value: 3F3Fh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| tx_rx_cell_max | 5:0 | RW | If the cell fill of the RX SAR output FIFO becomes greater than this value, cells from the TX SAR input FIFO will be blocked. 3Fh = no backpressure |
| tx_oa_cell_max | 9:6 | RW | If the cell fill of the port A output FIFO becomes greater than this value, cells from the TX SAR input FIFO will be blocked. Fh = no backpressure |
| tx_ob_cell_max | 13:10 | RW | If the cell fill of the port B output FIFO becomes greater than this value, cells from the TX SAR input FIFO will be blocked. Fh = no backpressure |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 154 - TX_SAR Overflow Register 0**

**Address: 69Ah**
**Label: tx_sar_overflow1**
**Reset Value: 000Fh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| tx_oc_cell_max | 3:0 | RW | If the cell fill of the port C output FIFO becomes greater than this value, cells from the TX SAR input FIFO will be blocked. Fh = no backpressure |
| reserved | 15:4 | RW | Reserved. Must always be “0000_0000_0000” |

**Table 155 - TX_SAR Overflow Register 1**

**Address: 6A0h**
**Label: porta_address**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| porta_add | 4:0 | RW | The address to which the chip will respond in UTOPIA level 2 addressing. |
| reserved | 15:5 | RW | Reserved. Must always be “0000_0000_000” |

**Table 156 - Port A Address Register**

**Address: 6A2h**
**Label: hec_byte_control**
**Reset Value: 0055h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| hec_mask | 7:0 | RW | The value by which the HEC byte will be XORed before being transmitted. |
| reserved | 15:8 | RW | Reserved. Must always be “0000_0000” |

**Table 157 - HEC Byte Control Register**

**Address: 6A4h**
**Label: unknown_header_routing**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rxa_ncr | 3:0 | RW | Routing of non-OAM cells from port A that fail the VPI-VCI match and mask test. “xxx1” = send to port A, “xx1x” = send to port B, “x1xx” = send to port C, “1xxx” = send to AAL0. |
| rxb_ncr | 7:4 | RW | Routing of non-OAM cells from port B that fail the VPI-VCI match and mask test. “xxx1” = send to port A, “xx1x” = send to port B, “x1xx” = send to port C, “1xxx” = send to AAL0. |
| rxc_ncr | 11:8 | RW | Routing of non-OAM cells from port C that fail the VPI-VCI match and mask test. “xxx1” = send to port A, “xx1x” = send to port B, “x1xx” = send to port C, “1xxx” = send to AAL0. |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 158 - Unknown Header Routing Register**

**Address: 6A6h**
**Label: unknown_oam_routing**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| rxa_ocr | 3:0 | RW | Routing of OAM cells from port A that fail the VPI-VCI match and mask test. “xxx1” = send to port A, “xx1x” = send to port B, “x1xx” = send to port C, “1xxx” = send to AAL0. |
| rxb_ocr | 7:4 | RW | Routing of OAM cells from port B that fail the VPI-VCI match and mask test. “xxx1” = send to port A, “xx1x” = send to port B, “x1xx” = send to port C, “1xxx” = send to AAL0. |
| rxc_ocr | 11:8 | RW | Routing of OAM cells from port C that fail the VPI-VCI match and mask test. “xxx1” = send to port A, “xx1x” = send to port B, “x1xx” = send to port C, “1xxx” = send to AAL0. |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 159 - Unknown OAM Routing Register**

### 3.7 H.100/H.110 Registers

**Address: 700h**
**Label: control**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| pll_clk_in_frequency | 1:0 | RW | “00”= 8.192 MHz; “01”= 16.384 MHz; “10”= 32.768 MHz;”11”= 65.536 MHz. |
| global_oe | 2 | RW | '0' = internal loopback (all OEs disabled); '1'=global OE enabled. |
| number_of_active_streams | 4:3 | RW | “00”= 32 active streams; “01” = 16 active streams; “10” = 8 active streams; “11” = 4 active streams. |
| h100_ct_c8_frame_loopback | 5 | RW | When '1', the ct_c8_X and ct_frame_X are looped back internally, without the need of going out on the actual pins. '0' for normal operation. |
| tdm_stream_format | 13:6 | RW | Each bit represents a quad of TDM streams. Bit 6 is quad 0. '0' = Format A; '1' = Format B. |
| reserved | 14 | RW | Reserved. Must always be '0'. |
| test_status | 15 | TS | Reserved. Must always be “0”. |

**Table 160 - H.100/H.110 Control Register**

**Address: 702h**
**Label: status0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mem_clk_alarm0 | 0 | ROL | Alarm that can be used to generate a programmable interval interrupt (me 9998888m_clk Alarm Timer 0). This bit is set when the mem_clk_count[31:16] matches the mem_clk_alarm0_count. Write a 1 to clear the bit. |
| mem_clk_alarm1 | 1 | ROL | Alarm that can be used to generate a programmable interval interrupt (mem_clk Alarm Timer 1). This bit is set when the mem_clk_count[31:16] matches the mem_clk_alarm1_count. Write a 1 to clear the bit. |
| ct_frame_a_bad | 2 | ROL | When the ct_frame_a is considered bad by the hardware, this bit is set. The frame is considered bad if it is not low one cycle out of 1024. Write a 1 to clear the bit. |
| ct_frame_b_bad | 3 | ROL | When the ct_frame_b is considered bad by the hardware, this bit is set. The frame is considered bad if it is not low one cycle out of 1024. Write a 1 to clear the bit. |
| ct_c8_a_bad | 4 | ROL | When the ct_c8_a is considered bad by the hardware, this bit is set. The clock is considered bad if its rising edges are too close or too far apart(i.e. if the rising edge does not arrive within 35 ns to the expected position). Write a 1 to clear the bit. |
| ct_c8_b_bad | 5 | ROL | When the ct_c8_b is considered bad by the hardware, this bit is set. The clock is considered bad if its rising edges are too close or too far apart(i.e. if the rising edge does not arrive within 35 ns to the expected position). Write a 1 to clear the bit. |
| adapa_point_lost | 6 | ROL | When a point is lost due to too much latency to the external memory, this bit is set. This is not a fatal error, but an indication that this phenomenon occurred. Point lost must be compensated for by software using the UUI field in each point. Write a 1 to clear the bit. |
| adapb_point_lost | 7 | ROL | When a point is lost due to too much latency to the external memory, this bit is set. This is not a fatal error, but an indication that this phenomenon occurred. Point lost must be compensated for by software using the UUI field in each point. Write a 1 to clear the bit. |
| tx_time_slot_pnt_out_of_sync | 8 | ROL | When this bit is high, the H.100 slave has lost its framing on the bus. This bit will cause the H.100 data pins to be tri-stated. Note that this bit will always be set at start-up. Write a 1 to clear the bit. |
| Reserved | 11:9 | ROL | Reserved. Must always be “000” |
| ct_frame_a_bad_rol | 12 | ROL | Same as bit 2, except this bit has no effect on the slaveship state. |
| ct_frame_b_bad_rol | 13 | ROL | Same as bit 3, except this bit has no effect on the slaveship state. |
| ct_c8_a_bad_rol | 14 | ROL | Same as bit 4, except this bit has no effect on the slaveship state. |
| ct_c8_b_bad_rol | 15 | ROL | Same as bit 5, except this bit has no effect on the slaveship state. |

**Table 161 - H.100/H.110 Status Register 0**

**Address: 704h**
**Label: status0_ie**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mem_clk_alarm0_ie | 0 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| mem_clk_alarm1_ie | 1 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ct_frame_a_bad_ie | 2 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ct_frame_b_bad_ie | 3 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ct_c8_a_bad_ie | 4 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ct_c8_b_bad_ie | 5 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| adapa_point_lost_ie | 6 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| adapb_point_lost_ie | 7 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| tx_time_slot_pnt_out_of_sync_ie | 8 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| Reserved | 11:9 | RO | Reserved. Must always be “000” |
| ct_frame_a_bad_rol_ie | 12 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ct_frame_b_bad_rol_ie | 13 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ct_c8_a_bad_rol_ie | 14 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ct_c8_b_bad_rol_ie | 15 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |

**Table 162 - H.100/H.110 Interrupt Enable Register 0**

**Address: 708h**
**Label: mem_clk_count0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mem_clk_count[31:16] | 15:0 | RO | Free-running mem_clk counter. |

**Table 163 - Memory Clock Counter 0**

**Address: 70Ah**
**Label: mem_clk_count1**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mem_clk_count[15:0] | 15:0 | RO | Free-running mem_clk counter. |

**Table 164 - Memory Clock Counter 1**

**Address: 70Ch**
**Label: mem_clk_alarm0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mem_clk_alarm0_count | 15:0 | RW | Alarm Timer 0 Value. When this value first matches the mem_clk_count[31:16], a status bit is set. |

**Table 165 - Memory Clock Alarm Register 0**

**Address: 70Eh**
**Label: mem_clk_alarm1**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mem_clk_alarm1_count | 15:0 | RW | Alarm Timer 1 Value. When this value first matches the mem_clk_count[31:16], a status bit is set. |

**Table 166 - Memory Clock Alarm Register 1**

**Address: 710h**
**Label: adapa0**
**Reset Value: 0001h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| Adapa_keep_one_pulse_out_of_x | 9:0 | RW | This field indicates how many clock recovery points should be received per point written to the point memory. Typically, this value would be set to 1. Range 1 to 1024. |
| Adapa_clk_divisor_reset | 10 | RW | When '0', this will force the Adaptive module A's clock divisor to reset. |
| Adapa_source | 12:11 | RW | “00” = clkrecov_pulse_a; “01” = gpio[0] (any change); “10” = gpio[0] (rising edge); “11” = gpio[0] (falling edge). |
| adapa_clk_divisor_load_now | 13 | PUL | When written to '1', this will allow the new div_integer and div_fraction to be used. |
| Reserved | 15:14 | PUL | Reserved. Always read as “00” |

**Table 167 - Adaptive Module A Register 0**

**Address: 712h**
**Label: adapa1**
**Reset Value: 0002h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| adapa_div_integer | 15:0 | RW | Adaptive module A's mem_clk divisor (integer part). Range 2 to FFFFh. |

**Table 168 - Adaptive Module A Register 1**

**Address: 714h**
**Label: adapa2**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| Adapa_div_fraction | 15:0 | RW | Adaptive module A's mem_clk divisor (fractional part). Range 0 to FFFFh. |

**Table 169 - Adaptive Module A Register 2**

**Address: 718h**
**Label: adapb0**
**Reset Value: 0001h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| Adapb_keep_one_pulse_out_of_x | 9:0 | RW | This field indicates how many clock recovery points should be received per point written to the point memory. Typically, this value would be set to 1. Range 1 to 1024. |
| Adapb_clk_divisor_reset | 10 | RW | When '0', this will force the Adaptive module B's clock divisor to reset. |
| Adapb_source | 12:11 | RW | “00” = clkrecov_pulse_b; “01” = gpio[1] (any change); “10” = gpio[1] (rising edge); “11” = gpio[1] (falling edge). |
| Adapb_clk_divisor_load_now | 13 | PUL | When written to '1', this will allow the new div_integer and div_fraction to be used. |
| Reserved | 15:14 | PUL | Reserved. Always read as “00” |

**Table 170 - Adaptive Module B Register 0**

**Address: 71Ah**
**Label: adapb1**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| Adapb_div_integer | 15:0 | RW | Adaptive module B's mem_clk divisor (integer part). Range 2 to FFFFh. |

**Table 171 - Adaptive Module B Register 1**

**Address: 71Ch**
**Label: adapb2**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| adapb_div_fraction | 15:0 | RW | Adaptive module B's mem_clk divisor (fractional part). Range 0 to FFFFh. |

**Table 172 - Adaptive Module B Register 2**

**Address: 720h**
**Label: mastership0**
**Reset Value: 0200h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ct_c8_frame_a_oe | 0 | RW | Controls the output enable of the ct_c8_a and ct_frame_a. '0' = tri-state; '1' = drive pins. |
| ct_c8_frame_b_oe | 1 | RW | Controls the output enable of the ct_c8_b and ct_frame_b. '0' = tri-state; '1' = drive pins. |
| ct_compatibility_oe | 2 | RW | Controls the output enable of the H100 compatibility signals. '0' = tri-state; '1' = drive pins. |
| reserved | 4:3 | RO | Reserved. Always read as “00” |
| fr_comp_polarity | 5 | RW | fr_comp pin polarity. '0' = Active low; '1' = active high. |
| fr_comp_type | 7:6 | RW | fr_comp pin timing type. “00”=straddle clock boundary; “01”= active during last bit; “10” = active during first bit; “11” = reserved. |
| fr_comp_frequency | 9:8 | RW | fr_comp frequency (pulse width). “00” = fr_comp related clock frequency at 2 MHz; “01” = fr_comp related clock frequency at 4 MHz; “10” = fr_comp related clock frequency at 8 MHz; “11”=reserved. |
| sclk_invert | 10 | RW | When '1', it inverts the sclk polarity as defined in the H100 Specifications. |
| sclkx2_invert | 11 | RW | When '1', it inverts the sclkx2 polarity as defined in the H100 Specifications. |
| sclk_frequency | 13:12 | RW | Selects the sclk frequency for the generation of sclk and sclk2. “00” = 2 MHz; “01” = 4 MHz; “10” = 8 MHz; “11” = reserved. |
| reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 173 - H.100/H.110 Master Register 0**

**Address: 722h**
**Label: mastership1**
**Reset Value: 0002h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| mastership_mode | 1:0 | RW | Selects the mode of the Master Circuitry. “00” = backup on A; “01” = backup on B; “10” = master on A; “11” = master on B. When ct_c8_frame_a_oe, ct_c8_frame_b_oe, ct_compatibility_oe are '0', this value does not matter. |
| slaveship_mode | 3:2 | RW | Selects the mode of the Slave Circuitry. “00” = Track ct_c8_a & ct_frame_a; “01” = Track ct_c8_b & ct_frame_b; “10” = Track ct_c8_a & ct_frame_a, but if they fail track ct_c8_b & ct_frame_b; “11” = Track ct_c8_b & ct_frame_b, but if they fail track ct_c8_a & ct_frame_a. |
| reserved | 15:4 | RW | Reserved. Must always be “0000_0000_0000” |

**Table 174 - H.100/H.110 Master Register 1**

**Address: 730h**
**Label: clock_rates**
**Reset Value: AAAAh**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| groupa_speed | 1:0 | RW | Selects the clock speed of H100 streams ct_c[3:0]. “00” = 2 MHz; “01” = 4 MHz; “10” = 8 MHz; “11” = reserved. |
| groupb_speed | 3:2 | RW | Selects the clock speed of H100 streams ct_c[7:4]. “00” = 2 MHz; “01” = 4 MHz; “10” = 8 MHz; “11” = reserved. |
| groupc_speed | 5:4 | RW | Selects the clock speed of H100 streams ct_c[11:8]. “00” = 2 MHz; “01” = 4 MHz; “10” = 8 MHz; “11” = reserved. |
| groupd_speed | 7:6 | RW | Selects the clock speed of H100 streams ct_c[15:12]. “00” = 2 MHz; “01” = 4 MHz; “10” = 8 MHz; “11” = reserved. |
| groupe_speed | 9:8 | RW | Selects the clock speed of H100 streams ct_c[19:16]. “00” = 2 MHz; “01” = 4 MHz; “10” = 8 MHz; “11” = reserved. |
| groupf_speed | 11:10 | RW | Selects the clock speed of H100 streams ct_c[23:20]. “00” = 2 MHz; “01” = 4 MHz; “10” = 8 MHz; “11” = reserved. |
| groupg_speed | 13:12 | RW | Selects the clock speed of H100 streams ct_c[27:24]. “00” = 2 MHz; “01” = 4 MHz; “10” = 8 MHz; “11” = reserved. |
| grouph_speed | 15:14 | RW | Selects the clock speed of H100 streams ct_c[31:28]. “00” = 2 MHz; “01” = 4 MHz; “10” = 8 MHz; “11” = reserved. |

**Table 175 - Clock Rates Register**

**Address: 732h**
**Label: timing_conf**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| slave_sampling | 1:0 | RW | Selects sampling point for the ct_d pins. The sampling point is relative to the ct_c8 clock period. “00” = falling edge; “01”= 75%; “10” = rising-edge; “11” = reserved. |
| oe_early_disable_override | 2 | RW | Selects if the ct_d output should be tri-stated before the rising edge of the ct_c8 clock or immediately after it. '0' = output enable asserted after rising edge and de-asserted before rising edge; '1' = output enable asserted and de-asserted after rising edge. |
| reserved | 15:3 | RW | Reserved. Must always be “0000_0000_0000_0” |

**Table 176 - Timing Configuration Register**

**Address: 740h**
**Label: gpio_out_reg0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| gpio_0_output_select | 4:0 | RW | Output mux selector. “00000”=ct_c8_a_in; “00001”=ct_c8_b_in; “00010”=ct_frame_a_in; “00011”=ct_frame_b_in; “00100”=ct_c8_selected; “00101”=ct_frame_selected; “00110”=output_constant ('0'/'1'); “00111”=ct_netref1_in; “01000”=ct_netref2_in; “01001”=gpio_in(0); “01010”=gpio_in(1); “01011”=gpio_in(2); “01100”=gpio_in(3); “01101”=gpio_in(4); “01110”=gpio_in(5); “01111”=gpio_in(6); “10000”=gpio_in(7); “10001”=adapa_ref; “10010”=adapb_ref; “10011”=clkrecov_pulse_a; “10100”=clkrecov_pulse_b; “10101”=mc_clock; “10110”=ct_mc_in. |
| gpio_0_output_constant | 5 | RW | If the gpio_X_output_select selects a constant, this value will be sent out on the gpio_X pin. |
| gpio_0_output_enable | 6 | RW | gpio_X pin output enable. '0'= tri-state; '1'= output enabled. |
| Reserved | 7 | RO | Reserved. Always read as “0” |
| gpio_1_output_select | 12:8 | RW | Output mux selector. “00000”=ct_c8_a_in; “00001”=ct_c8_b_in; 00010=ct_frame_a_in; “00011”=ct_frame_b_in; “00100”=ct_c8_selected; “00101”=ct_frame_selected; “00110”=output_constant ('0'/'1'); “00111”=ct_netref1_in; “01000”=ct_netref2_in; “01001”=gpio_in(0); “01010”=gpio_in(1); “01011”=gpio_in(2); “01100”=gpio_in(3); “01101”=gpio_in(4); “01110”=gpio_in(5); “01111”=gpio_in(6); “10000”=gpio_in(7); “10001”=adapa_ref; “10010”=adapb_ref; “10011”=clkrecov_pulse_a; “10100”=clkrecov_pulse_b; “10101”=mc_clock; “10110”=ct_mc_in. |
| gpio_1_output_constant | 13 | RW | If the gpio_X_output_select selects a constant, this value will be sent out on the gpio_X pin. |
| gpio_1_output_enable | 14 | RW | gpio_X pin output enable. '0'= tri-state; '1'= output enabled. |
| Reserved | 15 | RO | Reserved. Always read as “0” |

**Table 177 - General Purpose I/O Output Register 0**

**Address: 742h**
**Label: gpio_out_reg1**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| gpio_2_output_select | 4:0 | RW | Output mux selector. “00000”=ct_c8_a_in; “00001”=ct_c8_b_in; 00010=ct_frame_a_in; “00011”=ct_frame_b_in; “00100”=ct_c8_selected; “00101”=ct_frame_selected; “00110”=output_constant ('0'/'1'); “00111”=ct_netref1_in; “01000”=ct_netref2_in; “01001”=gpio_in(0); “01010”=gpio_in(1); “01011”=gpio_in(2); “01100”=gpio_in(3); “01101”=gpio_in(4); “01110”=gpio_in(5); “01111”=gpio_in(6); “10000”=gpio_in(7); “10001”=adapa_ref; “10010”=adapb_ref; “10011”=clkrecov_pulse_a; “10100”=clkrecov_pulse_b; “10101”=mc_clock; “10110”=ct_mc_in. |
| gpio_2_output_constant | 5 | RW | If the gpio_X_output_select selects a constant, this value will be sent out on the gpio_X pin. |
| gpio_2_output_enable | 6 | RW | gpio_X pin output enable. '0'= tri-state; '1'= output enabled. |
| reserved | 7 | RO | Reserved. Always read as “0” |
| gpio_3_output_select | 12:8 | RW | Output mux selector. “00000”=ct_c8_a_in; “00001”=ct_c8_b_in; 00010=ct_frame_a_in; “00011”=ct_frame_b_in; “00100”=ct_c8_selected; “00101”=ct_frame_selected; “00110”=output_constant ('0'/'1'); “00111”=ct_netref1_in; “01000”=ct_netref2_in; “01001”=gpio_in(0); “01010”=gpio_in(1); “01011”=gpio_in(2); “01100”=gpio_in(3); “01101”=gpio_in(4); “01110”=gpio_in(5); “01111”=gpio_in(6); “10000”=gpio_in(7); “10001”=adapa_ref; “10010”=adapb_ref; “10011”=clkrecov_pulse_a; “10100”=clkrecov_pulse_b; “10101”=mc_clock; “10110”=ct_mc_in. |
| gpio_3_output_constant | 13 | RW | If the gpio_X_output_select selects a constant, this value will be sent out on the gpio_X pin. |
| gpio_3_output_enable | 14 | RW | gpio_X pin output enable. '0'= tri-state; '1'= output enabled. |
| reserved | 15 | RO | Reserved. Always read as “0” |

**Table 178 - General Purpose I/O Output Register 1**

**Address: 744h**
**Label: gpio_out_reg2**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| gpio_4_output_select | 4:0 | RW | Output mux selector. “00000”=ct_c8_a_in; “00001”=ct_c8_b_in; 00010=ct_frame_a_in; “00011”=ct_frame_b_in; “00100”=ct_c8_selected; “00101”=ct_frame_selected; “00110”=output_constant ('0'/'1'); “00111”=ct_netref1_in; “01000”=ct_netref2_in; “01001”=gpio_in(0); “01010”=gpio_in(1); “01011”=gpio_in(2); “01100”=gpio_in(3); “01101”=gpio_in(4); “01110”=gpio_in(5); “01111”=gpio_in(6); “10000”=gpio_in(7); “10001”=adapa_ref; “10010”=adapb_ref; “10011”=clkrecov_pulse_a; “10100”=clkrecov_pulse_b; “10101”=mc_clock; “10110”=ct_mc_in |
| gpio_4_output_constant | 5 | RW | If the gpio_X_output_select selects a constant, this value will be sent out on the gpio_X pin. |
| gpio_4_output_enable | 6 | RW | gpio_X pin output enable. '0'= tri-state; '1'= output enabled. |
| reserved | 7 | RO | Reserved. Always read as “0” |
| gpio_5_output_select | 12:8 | RW | Output mux selector. “00000”=ct_c8_a_in; “00001”=ct_c8_b_in; 00010=ct_frame_a_in; “00011”=ct_frame_b_in; “00100”=ct_c8_selected; “00101”=ct_frame_selected; “00110”=output_constant ('0'/'1'); “00111”=ct_netref1_in; “01000”=ct_netref2_in; “01001”=gpio_in(0); “01010”=gpio_in(1); “01011”=gpio_in(2); “01100”=gpio_in(3); “01101”=gpio_in(4); “01110”=gpio_in(5); “01111”=gpio_in(6); “10000”=gpio_in(7); “10001”=adapa_ref; “10010”=adapb_ref; “10011”=clkrecov_pulse_a; “10100”=clkrecov_pulse_b; “10101”=mc_clock; “10110”=ct_mc_in |
| gpio_5_output_constant | 13 | RW | If the gpio_X_output_select selects a constant, this value will be sent out on the gpio_X pin. |
| gpio_5_output_enable | 14 | RW | gpio_X pin output enable. '0'= tri-state; '1'= output enabled. |
| reserved | 15 | RO | Reserved. Always read as “0” |

**Table 179 - General Purpose I/O Output Register 2**

**Address: 746h**
**Label: gpio_out_reg3**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| gpio_6_output_select | 4:0 | RW | Output mux selector. “00000”=ct_c8_a_in; “00001”=ct_c8_b_in; 00010=ct_frame_a_in; “00011”=ct_frame_b_in; “00100”=ct_c8_selected; “00101”=ct_frame_selected; “00110”=output_constant ('0'/'1'); “00111”=ct_netref1_in; “01000”=ct_netref2_in; “01001”=gpio_in(0); “01010”=gpio_in(1); “01011”=gpio_in(2); “01100”=gpio_in(3); “01101”=gpio_in(4); “01110”=gpio_in(5); “01111”=gpio_in(6); “10000”=gpio_in(7); “10001”=adapa_ref; “10010”=adapb_ref; “10011”=clkrecov_pulse_a; “10100”=clkrecov_pulse_b; “10101”=mc_clock; “10110”=ct_mc_in |

**Table 180 - General Purpose I/O Output Register 3**

| | | | |
|---|---|---|---|
| gpio_6_output_constant | 5 | RW | If the gpio_X_output_select selects a constant, this value will be sent out on the gpio_X pin. |
| gpio_6_output_enable | 6 | RW | gpio_X pin output enable. '0'= tri-state; '1'= output enabled. |
| reserved | 7 | RO | Reserved. Always read as “0” |
| gpio_7_output_select | 12:8 | RW | Output mux selector. “00000”=ct_c8_a_in; “00001”=ct_c8_b_in; 00010=ct_frame_a_in; “00011”=ct_frame_b_in; “00100”=ct_c8_selected; “00101”=ct_frame_selected; “00110”=output_constant ('0'/'1'); “00111”=ct_netref1_in; “01000”=ct_netref2_in; “01001”=gpio_in(0); “01010”=gpio_in(1); “01011”=gpio_in(2); “01100”=gpio_in(3); “01101”=gpio_in(4); “01110”=gpio_in(5); “01111”=gpio_in(6); “10000”=gpio_in(7); “10001”=adapa_ref; “10010”=adapb_ref; “10011”=clkrecov_pulse_a; “10100”=clkrecov_pulse_b; “10101”=mc_clock; “10110”=ct_mc_in |
| gpio_7_output_constant | 13 | RW | If the gpio_X_output_select selects a constant, this value will be sent out on the gpio_X pin. |
| gpio_7_output_enable | 14 | RW | gpio_X pin output enable. '0'= tri-state; '1'= output enabled. |
| reserved | 15 | RO | Reserved. Always read as “0” |

**Table 180 - General Purpose I/O Output Register 3 (continued)**

**Address: 748h**
**Label: gpio_out_reg4**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| ct_netref1_output_select | 4:0 | RW | Output mux selector. “00000”=ct_c8_a_in; “00001”=ct_c8_b_in; 00010=ct_frame_a_in; “00011”=ct_frame_b_in; “00100”=ct_c8_selected; “00101”=ct_frame_selected; “00110”=output_constant ('0'/'1'); “00111”=ct_netref1_in; “01000”=ct_netref2_in; “01001”=gpio_in(0); “01010”=gpio_in(1); “01011”=gpio_in(2); “01100”=gpio_in(3); “01101”=gpio_in(4); “01110”=gpio_in(5); “01111”=gpio_in(6); “10000”=gpio_in(7); “10001”=adapa_ref; “10010”=adapb_ref; “10011”=clkrecov_pulse_a; “10100”=clkrecov_pulse_b; “10101”=mc_clock; “10110”=ct_mc_in. |
| ct_netref1_output_constant | 5 | RW | If the ct_netrefX_select selects a constant, this value will be sent out on the ct_netrefX pin. |
| ct_netref1_output_enable | 6 | RW | ct_netrefX pin output enable. '0'= tri-state; '1'= output enabled. |
| reserved | 7 | RO | Reserved. Always read as “0” |
| ct_netref2_output_select | 12:8 | RW | Output mux selector. “00000”=ct_c8_a_in; “00001”=ct_c8_b_in; 00010=ct_frame_a_in; “00011”=ct_frame_b_in; “00100”=ct_c8_selected; “00101”=ct_frame_selected; “00110”=output_constant ('0'/'1'); “00111”=ct_netref1_in; “01000”=ct_netref2_in; “01001”=gpio_in(0); “01010”=gpio_in(1); “01011”=gpio_in(2); “01100”=gpio_in(3); “01101”=gpio_in(4); “01110”=gpio_in(5); “01111”=gpio_in(6); “10000”=gpio_in(7); “10001”=adapa_ref; “10010”=adapb_ref; “10011”=clkrecov_pulse_a; “10100”=clkrecov_pulse_b; “10101”=mc_clock; “10110”=ct_mc_in |

**Table 181 - General Purpose I/O Output Register 4**

| | | | |
|---|---|---|---|
| ct_netref2_output_constant | 13 | RW | If the ct_netrefX_select selects a constant, this value will be sent out on the ct_netrefX pin. |
| ct_netref2_output_enable | 14 | RW | ct_netrefX pin output enable. '0'= tri-state; '1'= output enabled. |
| reserved | 15 | RO | Reserved. Always read as “0” |

**Table 181 - General Purpose I/O Output Register 4 (continued)**

**Address: 74Ah**
**Label: gpio_in**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| gpio_0_value | 0 | RO | Current level of the corresponding pin. |
| gpio_1_value | 1 | RO | Current level of the corresponding pin. |
| gpio_2_value | 2 | RO | Current level of the corresponding pin. |
| gpio_3_value | 3 | RO | Current level of the corresponding pin. |
| gpio_4_value | 4 | RO | Current level of the corresponding pin. |
| gpio_5_value | 5 | RO | Current level of the corresponding pin. |
| gpio_6_value | 6 | RO | Current level of the corresponding pin. |
| gpio_7_value | 7 | RO | Current level of the corresponding pin. |
| ct_netref1_value | 8 | RO | Current level of the corresponding pin. |
| ct_netref2_value | 9 | RO | Current level of the corresponding pin. |
| reserved | 15:10 | RO | Reserved. Always read as “0000_00” |

**Table 182 - General Purpose I/O Input Register**

**Address: 74Ch**
**Label: gpio_status**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| gpio_0_change | 0 | ROL | This bit is set if the level of the corresponding pin changes. |
| gpio_1_change | 1 | ROL | This bit is set if the level of the corresponding pin changes. |
| gpio_2_change | 2 | ROL | This bit is set if the level of the corresponding pin changes. |
| gpio_3_change | 3 | ROL | This bit is set if the level of the corresponding pin changes. |
| gpio_4_change | 4 | ROL | This bit is set if the level of the corresponding pin changes. |
| gpio_5_change | 5 | ROL | This bit is set if the level of the corresponding pin changes. |
| gpio_6_change | 6 | ROL | This bit is set if the level of the corresponding pin changes. |
| gpio_7_change | 7 | ROL | This bit is set if the level of the corresponding pin changes. |
| ct_netref1_change | 8 | ROL | This bit is set if the level of the corresponding pin changes. |

**Table 183 - General Purpose I/O Status Register**

| | | | |
|---|---|---|---|
| ct_netref2_change | 9 | ROL | This bit is set if the level of the corresponding pin changes. |
| reserved | 15:10 | ROL | Reserved. Always read as “0000_00” |

**Table 183 - General Purpose I/O Status Register (continued)**

**Address: 74Eh**
**Label: gpio_status_ie**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| gpio_0_change_ie | 0 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| gpio_1_change_ie | 1 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| gpio_2_change_ie | 2 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| gpio_3_change_ie | 3 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| gpio_4_change_ie | 4 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| gpio_5_change_ie | 5 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| gpio_6_change_ie | 6 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| gpio_7_change_ie | 7 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ct_netref1_change_ie | 8 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| ct_netref2_change_ie | 9 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| reserved | 15:10 | IE | Reserved. Always read as “0000_00” |

**Table 184 - General Purpose I/O Status Interrupt Enable Register**

**Address: 774h**
**Label: mastership_hidden0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| automatic_master_override | 0 | RW | When '1', the automatic H100 Master programming values are overridden. |
| mux0_select_override | 1 | RW | 0' = don't swap; '1' = swap. |
| mux1_select_override | 2 | RW | 0'=Divide by 4; '1' = ct_c8 pin. |
| mux2_select_override | 3 | RW | 0'=Local Ref; '1' = ct_c8 pin. |
| mux3_select_override | 5:4 | RW | “00”=PLL Used; “01”=32 MHz local ref used; “10”=16 MHz local ref used; “11”=8 MHz local ref used. |
| reserved | 15:6 | RW | Reserved. Always read as “0000_0000_00” |

**Table 185 - H.100/H.110 Master Hidden Register 0**

**Address: 776h**
**Label: mastership_hidden1**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| h100_check_automatic_override | 0 | WO | When '1', the following two field will control the ct_c8 frequency checker. |
| h100_min_mem_clk_ct_c8_override | 5:1 | RW | minimum period (in mem_clk cycles) between 2 H100 ct_c8 clock rising edges. |
| h100_max_mem_clk_ct_c8_override | 10:6 | RW | Maximum period (in mem_clk cycles) between 2 H100 ct_c8 clock rising edges. |
| h100_sample_tristate_override | 11 | WO | When '1', the values in the following registers will be used to control the sampling and tri-state delays. |
| Reserved | 15:12 | WO | Reserved. Always read as “0000” |

**Table 186 - H.100/H.110 Master Hidden Register 1**

**Address: 778h**
**Label: mastership_hidden2**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| h100_samp_clk_delay_flops | 3:0 | RW | Number of flops used = 8 + value of this register. “1111” is reserved for selecting falling edge ct_c8 clock |
| h100_samp_clk_delay_buff | 7:4 | RW | “1111” is reserved for selecting rising edge ct_c8 clock |
| h100_oe_clk_delay_flops | 11:8 | RW | Number of flops used = 8 + value of this register |
| h100_oe_clk_delay_buff | 15:12 | RW | |

**Table 187 - H.100/H.110 Master Hidden Register 2**

### 3.8 Miscellaneous Registers

**Address: 802h**
**Label: status0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cell_assembly_event_buf_overflow | 0 | ROL | Overflow in the cell assembly event queue. Fatal chip error. |
| pointa_buf_overflow | 1 | ROL | Overflow in the clock recovery point A buffer. |
| pointb_buf_overflow | 2 | ROL | Overflow in the clock recovery point B buffer. |
| reserved | 15:3 | ROL | Reserved. Always read as “0000_0000_0000_0” |

**Table 188 - Miscellaneous Status Register**

**Address: 804h**
**Label: status0_ie**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cell_assembly_event_buf_overflow | 0 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| pointa_buf_overflow | 1 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| pointb_buf_overflow | 2 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| reserved | 15:3 | RO | Reserved. Always read as “0000_0000_0000_0” |

**Table 189 - Miscellaneous Interrupt Enable Register**

**Address: 810h**
**Label: tone_buffer_control**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| sdram_tone_base | 4:0 | RW | Represents bits 23:19 of the byte address to the SDRAM tone buffers |
| sdram_tone_size | 6:5 | RW | “00” = 16KBytes, “01” = 32K, “10” = 64K, “11” = reserved. Size is for a single buffer. |
| sdram_tone_enable | 7 | RW | Enables the use of the tones contained in the SDRAM. |
| ssram_tone_enable | 8 | RW | Enables the use of the tones contained in the SSRAM. |
| reserved | 15:9 | RW | Reserved. Must always be “0000_000” |

**Table 190 - Tone Buffer Control Register**

**Address: 820h**
**Label: pointa_manage**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| pointa_buffer_base_add | 8:0 | RW | Bits 20:12 of the base address of the clock recovery point buffer A in external SSRAM |
| pointa_buffer_size | 11:9 | RW | “000” = 4KBytes, “001” = 8K, “010” = 16K, “011” = 32K, “100” = 64K, “101” = 128K, others = reserved. Size is for a single buffer. |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 191 - Point A Buffer Management Register**

**Address: 822h**
**Label: pointa_read**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| pointa_rpnt | 13:0 | RW | The CUP’s read pointer to the clock recovery point FIFO. This read pointer has an extra bit to allow for a full buffer (for example, 2048 points instead of 2047) |
| Reserved | 15:14 | RW | Reserved. Must always be “00” |

**Table 192 - Point A Read Pointer Register**

**Address: 824h**
**Label: pointa_write**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| pointa_wpnt | 13:0 | RO | The chip's write pointer to the clock recovery point FIFO. This read pointer has an extra bit to allow for a full buffer (for example, 2048 points instead of 2047) |
| Reserved | 15:14 | RO | Reserved. Always read as “00” |

**Table 193 - Point A Write Pointer Register**

**Address: 828h**
**Label: pointb_manage**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| pointb_buffer_base_add | 8:0 | RW | Bits 20:12 of the base address of the clock recovery point buffer B in external SSRAM |
| pointb_buffer_size | 11:9 | RW | “000” = 4KBytes, “001” = 8K, “010” = 16K, “011” = 32K, “100” = 64K, “101” = 128K, others = reserved |
| reserved | 15:12 | RW | Reserved. Must always be “0000” |

**Table 194 - Point B Buffer Management Register**

**Address: 82Ah**
**Label: pointb_read**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| pointb_rpnt | 13:0 | RW | The CUP's read pointer to the clock recovery point FIFO. This read pointer has an extra bit to allow for a full buffer (for example, 2048 points instead of 2047) |
| reserved | 15:14 | RW | Reserved. Must always be "00" |

**Table 195 - Point B Read Pointer Register**

**Address: 82Ch**
**Label: pointb_write**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| pointb_wpnt | 13:0 | RO | The chip's write pointer to the clock recovery point FIFO. This read pointer has an extra bit to allow for a full buffer (for example, 2048 points instead of 2047) |
| reserved | 15:14 | RO | Reserved. Always read as "00" |

**Table 196 - Point B Write Pointer Register**

**Address: 830h**
**Label: cid_base**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| cid_descriptor_base_address | 4:0 | RW | Bits 23:21 of the CID descriptor's base address in the SDRAM |
| reserved | 15:5 | RW | Reserved. Must always be "0000_0000_000" |

**Table 197 - CID Base Address Register**

### 3.9 RX TDM Registers

**Address: 900h**
**Label: control**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| hdlc_type | 0 | RW | 0'=Bit wise HDLC Framing; '1'=Byte Wise HDLC Framing. |
| rx_tdm_time_slot_lead | 3:1 | RW | Set to “111” for normal operation. |
| number_packets_per_frame | 6:4 | RW | “000” = 32 packets, “001” = 64 packets, “010” = 128 packets, “011” = 256 packets, “100” = 512 packets, “101” = 1024, “11x” = reserved |
| reserved | 7 | RW | Reserved. Must always be “0”. |
| llman_process_enable | 8 | RW | Enables the treatment of incoming RX SAR bytes going to the H100 bus. |
| u_law_zero_illegal | 9 | RW | When '1', the u-law value 00h will be replaced by 02h. |
| padding_law | 11:10 | RW | “00” = don't translate padding, OL bit in Rx TDM Control Structure is ignored for padding bytes; “01” = padding is in u-law only, “10” = padding is in A-law only. |
| test_status | 15 | TS | Reserved. Must always be “0”. |

**Table 198 - RX TDM Control Register**

**Address: 902h**
**Label: status0**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| llman_process_crashed | 0 | ROL | Indicates that the linked-list manager crashed. This indicates corrupt data in the linked list memory. |
| underrun_detect | 1 | ROL | Underrun detected on a channel. |
| underrun_wrap_detect | 2 | ROL | Indicates that the 16-bit underrun counter on a single channel has wrapped. |
| rxtdm_read_cache_overflow | 3 | ROL | Overflow in the RX TDM read cache. Fatal chip error. |
| rxtdm_write_cache_overflow | 4 | ROL | Overflow in the RX TDM write cache. Fatal chip error. |
| write_back_cache_overflow | 5 | ROL | Overflow in the RX TDM write back cache. |
| rx_tdm_fifo_overflow | 6 | ROL | Overflow in the RX TDM data FIFO. Fatal chip error. |
| rxsar_pointer_wb_overflow | 7 | ROL | Overflow in the RX SAR pointer write back FIFO. Fatal chip error. |
| reserved | 15:8 | ROL | Reserved. Always read as “0000_0000” |

**Table 199 - RX TDM Status Register 0**

**Address: 904h**
**Label: status0_ie**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| llman_process_crashed_ie | 0 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| underrun_detect_ie | 1 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| underrun_wrap_detect_ie | 2 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| rxtdm_read_cache_overflow_ie | 3 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| rxtdm_write_cache_overflow_ie | 4 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| write_back_cache_overflow_ie | 5 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| rx_tdm_fifo_overflow_ie | 6 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| rxsar_pointer_wb_overflow_ie | 7 | IE | When ‘1’ and the corresponding status bit is ‘1’ an interrupt will be generated. |
| reserved | 15:8 | RO | Reserved. Always read as “0000_0000” |

**Table 200 - RX TDM Interrupt Enable Register 0**

**Address: 908h**
**Label: monitor**
**Reset Value: 0000h**

| Label | Bit Position | Type | Description |
|---|---|---|---|
| hdlc_pcm_channel_error | 9:0 | RO | Indicates the channel number, according to its position in the RX TDM control memory, on which the last underrun or underrun wrap error occurred |
| reserved | 15:10 | RO | Reserved. Always read as “0000_00” |

**Table 201 - RX TDM Channel Number Monitor Register**

# 4.0 Electrical Specification

## 4.1 DC Characteristics

### Absolute Maximum Ratings

| | Parameter | Symbol | Min | Max | Units |
|---|---|---|---|---|---|
| | Supply Voltage - 3.3 Volt Rail | $V_{DD}$ | -0.5 | 5.0 | V |
| | Voltage on 3.3V Input pins | $V_I$ | -0.5 | VDD+0.5 | V |
| | Voltage on 3.3V Output pins | $V_O$ | -0.5 | VDD+0.5 | V |
| | Storage Temperature | $T_S$ | -40.0 | +125.0 | °C |

* Exceeding these figures may cause permanent damage. Functional operation under these conditions is not guaranteed. Voltage measurements are with respect to ground ($V_{SS}$) unless otherwise stated. Long-term exposure to absolute maximum ratings may affect device reliability, and permanent damage may occur if operate exceeding the rating. The device should be operated under recommended operating condition.

### Recommended Operating Conditions

| | Characteristics | Sym | Min | Typ[a] | Max | Units | Test Conditions |
|---|---|---|---|---|---|---|---|
| | Operating Temperature | $T_{OP}$ | 0.0 | 25.0 | 70.0 | °C | |
| | Supply Voltage, 3.3 Volt Rail | $V_{DD}$ | 3.135 | 3.3 | 3.465 | V | |

a. Typical figures are at 25°C and are for design aid only; not guaranteed and not subject to production testing. Voltage measurements are with respect to ground ($V_{SS}$) unless otherwise stated.

### DC Characteristics

| | Characteristics | Sym | Min | Typ[a] | Max | Units | Test Conditions [b] |
|---|---|---|---|---|---|---|---|
| | Supply Current - 3.3 V supply | $I_{DD}$ | | 1.19 | | A | 60.0 MHz, Nominal output loads, 1023 Channels |
| | Continuous current at digital inputs | $I_I$ | | 2.31 | | mA | |
| | Continuous current at digital outputs | $I_O$ | | 3.04 | | mA | |
| | Device Power Dissipation | $P_{DDS}$ | | 3.927 | | W | 60.0 MHz, Nominal output loads, 1023 Channels |
| | Input High Voltage | $V_{IH}$ | 0.7VDD | | | | |
| | Input Low Voltage | $V_{IL}$ | | | 0.3VDD | V | |
| | Output HIGH Voltage | $V_{OH}$ | 2.4 | | | V | |
| | Output LOW Voltage | $V_{OL}$ | | | 0.4 | V | |
| | Schmitt Trigger Positive Threshold | $V_{t+}$ | | 1.6 | | V | |
| | Schmitt Trigger Neg. Threshold | $V_{t-}$ | | 1.2 | | V | |
| | Input Leakage Current Inputs with gated pull-up Inputs with gated pull-down | $I_I$ | -1.0 | | 1.0 | µA | $V_{IN} = V_{DDx}$ or $V_{ss}$ |
| | | $I_{IH}$ | | -30.0 | -70.0 | µA | $V_{IN} = V_{DD}$ |
| | | $I_{IL}$ | | 30.0 | 70.0 | µA | $V_{IN} = V_{ss}$ |

## DC Characteristics (continued)

| | Characteristics | Sym | Min | Typ[a] | Max | Units | Test Conditions [b] |
|---|---|---|---|---|---|---|---|
| | Input Pin Capacitance | $C_I$ | | 5.0 | | pF | |
| | Output Pin Capacitance | $C_O$ | | 5.0 | | pF | |
| | Output High Impedance Leakage | $I_{OZ}$ | | | 1.0 | uA | $V_O = V_{SS}$ or $V_{DD}$ |
| | 3.3V output HIGH current (6 mA buffer) | $I_{OH}$ | | | 6.0 | mA | |
| | 3.3V output LOW current (6 mA buffer) | $I_{OL}$ | | | 6.0 | mA | |
| | 3.3V output HIGH current (12 mA buffer) | $I_{OH}$ | | | 12.0 | mA | |
| | 3.3V output LOW current (12 mA buffer) | $I_{OL}$ | | | 12.0 | mA | |
| | 3.3V output HIGH current (24 mA buffer) | $I_{OH}$ | | | 24.0 | mA | |
| | 3.3V output LOW current (24 mA buffer) | $I_{OL}$ | | | 24.0 | mA | |
| | Junction-to-Ambient Thermal Resistance | $\theta_{J-A}$ | | 14.0 | | °C/W | 0 cfm air flow (natural convection airflow only) |

a. Typical figures are at 25°C and are for design aid only; not guaranteed and not subject to production testing.
b. $T_{OP}$ = 0°C to 70°C; 3.135V $\leq V_{DD} \leq$ 3.465V
Voltage measurements are with respect to ground ($V_{SS}$) unless otherwise stated.

### 4.2 AC Characteristics

### 4.3 Intel/Motorola Interface

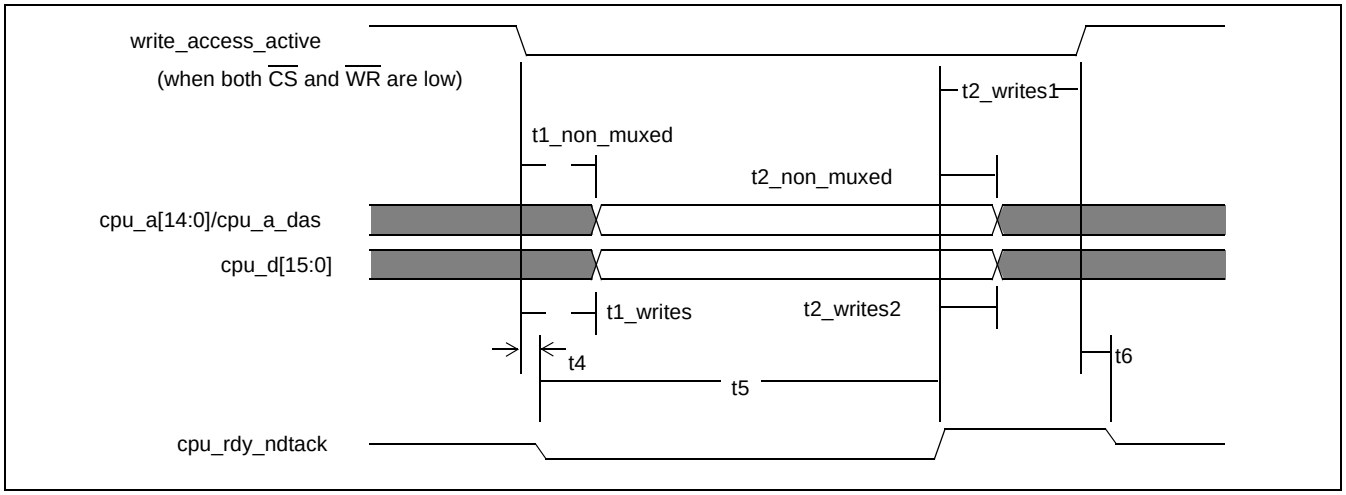



**Figure 71 - Non-Multiplexed CPU Interface - Intel Mode - Write Access**

| Symbol | Description | Min | Typical | Max | Unit |
|---|---|---|---|---|---|
| t1_non_muxed | write_access_active falling edge to cpu_a valid<br>cpu_a_das valid | | | 2 * upclk - 4 | ns |
| t1_writes | write_access_active falling edge to cpu_d valid | | | 2 * upclk - 4 | ns |
| t2_non_muxed | cpu_rdy_ndtack rising edge to cpu_a invalid<br>cpu_a_das invalid | 0 | | | ns |
| t2_writes1 | cpu_rdy_ndtack rising edge to write_access_active rising edge | 0 | | | ns |
| t2_writes2 | cpu_rdy_ndtack rising edge to cpu_d invalid | 0 | | | ns |
| t4 | write_access_active falling edge to cpu_rdy_ndtack falling edge | 0 | | 12 | ns |
| t5 | Write Access Time | | | 740 | ns |
| t6 | write_access_active rising edge to cpu_rdy_ndtack tri-state | 0 | | 10 | ns |

**Table 202 - Non-Multiplexed CPU Interface - Intel Mode - Write Access**

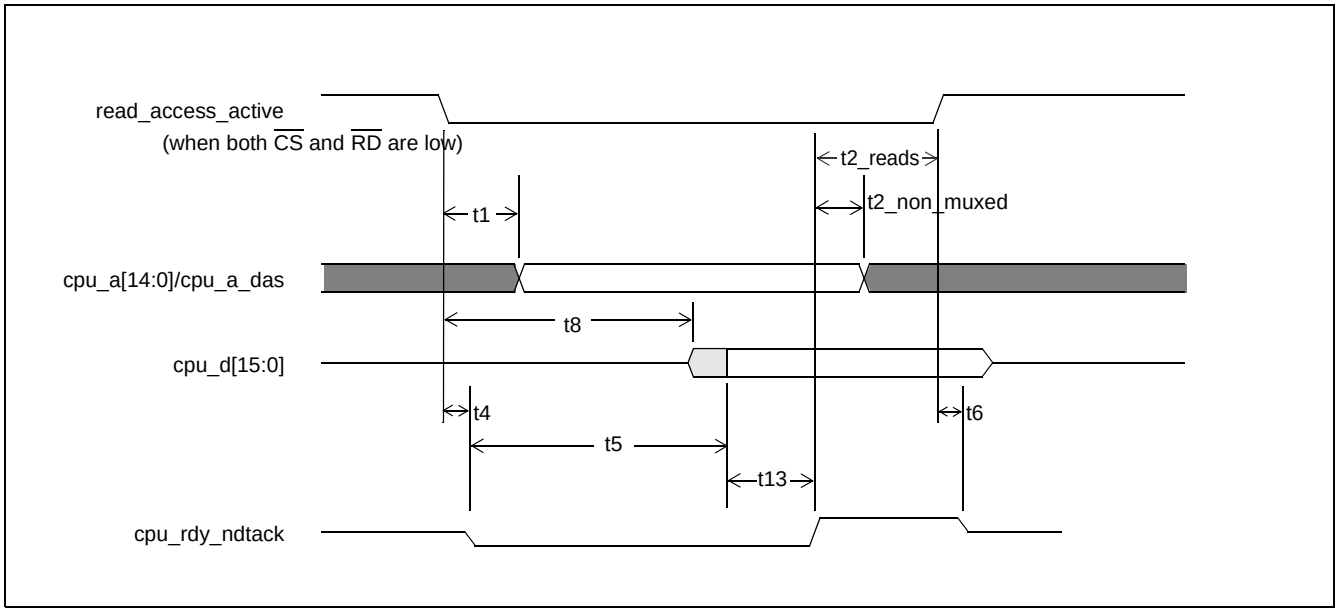



**Figure 72 - Non-Multiplexed CPU Interface - Intel Mode - Read Access**

| Symbol | Description | Min | Typical | Max | Unit |
|---|---|---|---|---|---|
| t1 | read_access_active falling edge to cpu_a valid cpu_a_das valid | | | 2 * upclk - 4 | ns |
| t2_reads | cpu_rdy_ndtack rising edge to read_access_active rising edge (reads) | 0 | | | ns |
| t2_non_muxed | cpu_rdy_ndtack rising edge to cpu_a invalid (non-multiplexed) cpu_a_das invalid (non-multiplexed) | | | | ns |
| t4 | read_access_active falling edge to cpu_rdy_ndtack falling edge | 0 | | 12 | ns |
| t5 | Read Access Time | | | See Table 205, “t5 Read Access Times,” on page 177. | |
| t6 | read_access_active rising edge to cpu_rdy_ndtack tri-state | 0 | | 10 | ns |
| t8 | read_access_active falling edge to cpu_d driven | 3 * upclk - 4 | | | ns |
| t13 | cpu_d valid to cpu_rdy_ndtack rising edge | upclk - 4 | | | ns |

**Table 203 - Non-Multiplexed CPU Interface - Intel Mode - Read Access**

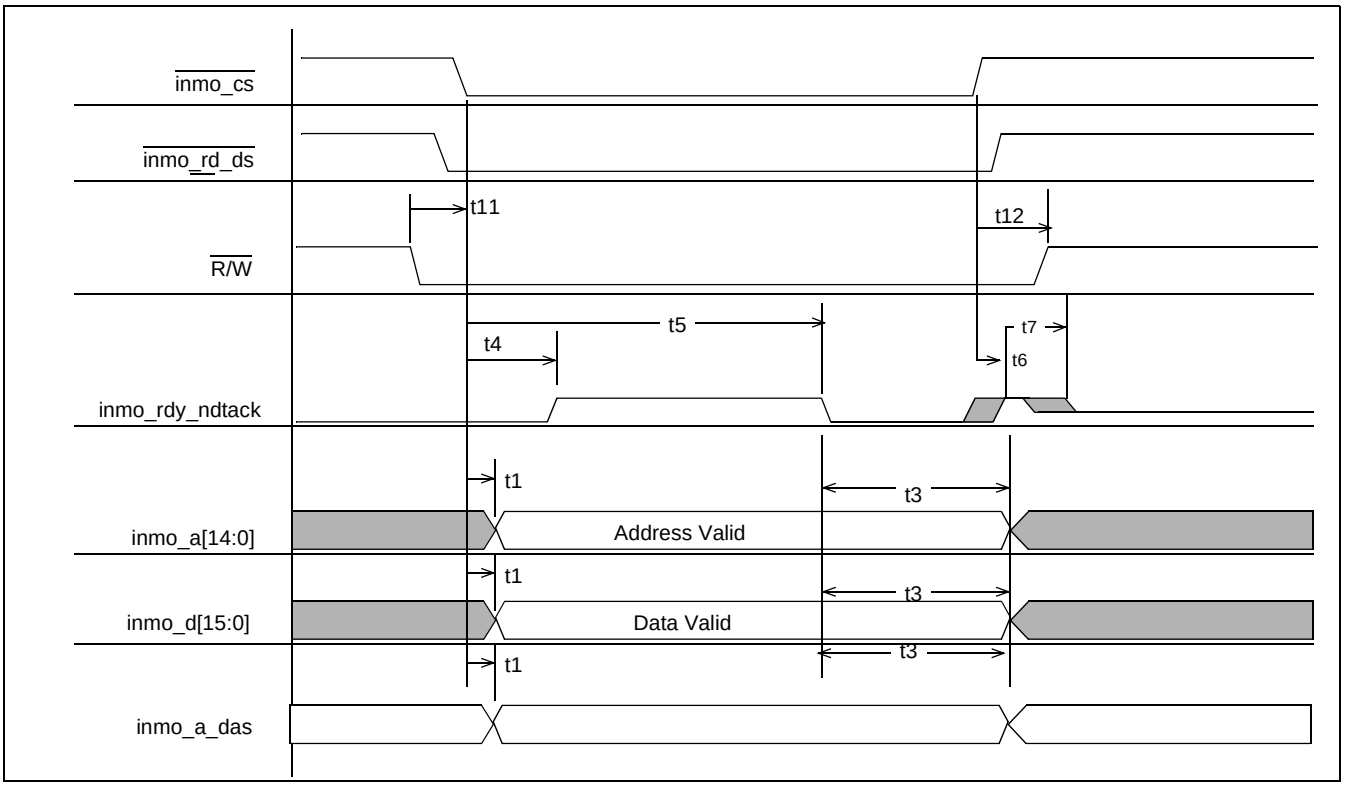



**Figure 73 - Non-Multiplexed CPU Interface - Motorola Mode - Write Access**

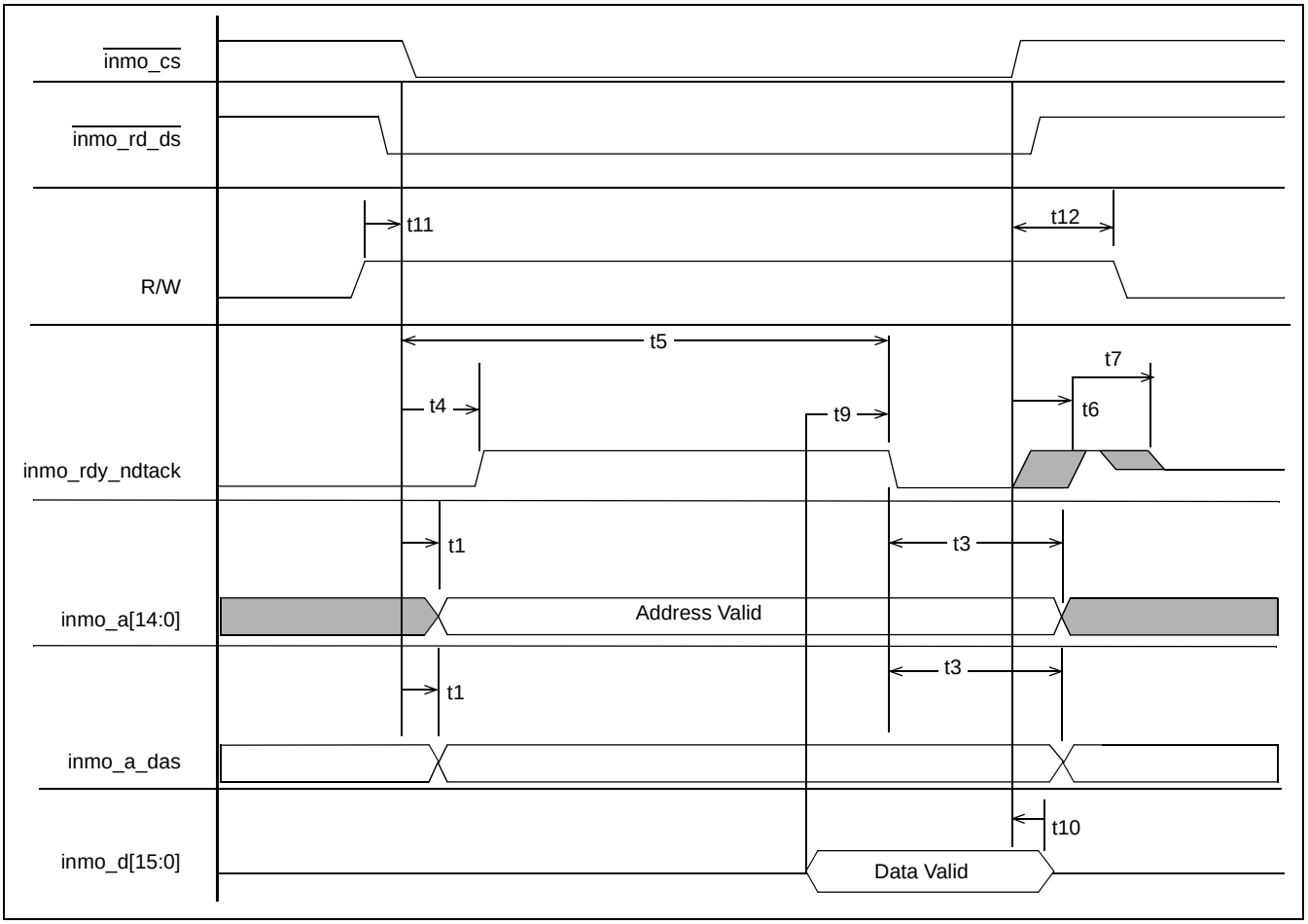



**Figure 74 - Non-Multiplexed CPU Interface - Motorola Mode - Read Access**

| Sym | Description Write Access | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| t1 | Address & Data Setup -- $\overline{\text{cpu_cs}}$ and $\overline{\text{cpu_rd_ds}}$ asserted to cpu_a[14:0] and cpu_d[15:0] and cpu_a_das valid | | | 2*upclk - 4 | ns |
| t3 | Address & Data Hold -- cpu_rdy_ndtack low to cpu_a[14:0] and cpu_d[15:0] and cpu_a_das invalid | 0 | | | ns |
| t4 | cpu_rdy_ndtack high -- $\overline{\text{cpu_cs}}$ and $\overline{\text{cpu_rd_ds}}$ asserted to cpu_rdy_ndtack driving one | 0 | | 12 | ns |
| t5 | cpu_rdy_ndtack delay -- $\overline{\text{cpu_cs}}$ and $\overline{\text{cpu_rd_ds}}$ asserted to cpu_rdy_ndtack driving zero | | | 740 | ns |
| t6 | cpu_rdy_ndtack hold -- $\overline{\text{cpu_cs}}$ and $\overline{\text{cpu_rd_ds}}$ de-asserted to cpu_rdy_ndtack driving one | 0 | | 10 | ns |
| t7 | cpu_rdy_ndtack high impedance -- cpu_rdy_ndtack driving one to cpu_rdy_ndtack high impedance | 2 | | 8 | ns |
| t11 | cpu_r/$\overline{\text{w}}$ low to both $\overline{\text{cpu_cs}}$ and $\overline{\text{cpu_ds}}$ asserted | 0 | | | ns |
| t12 | $\overline{\text{cpu_cs}}$ or $\overline{\text{cpu_ds}}$ high to cpu_r/$\overline{\text{w}}$ high | 0 | | | ns |
| Note: t1, t4, and t5 are dependent upon the last of $\overline{\text{cpu_cs}}$ and $\overline{\text{cpu_rd_ds}}$ to be asserted. t6 is dependent on the first of cpu_cs and cpu_rd_ds to be de-asserted. | | | | | |
| **Sym** | **Description Read Access** | **Min** | **Typ** | **Max** | **Unit** |
| t1 | Address Setup -- $\overline{\text{cpu_cs}}$ and $\overline{\text{cpu_rd_ds}}$ asserted to cpu_a[14:0] and cpu_a_das valid | | | 2*upclk - 4 | ns |
| t3 | Address Hold -- cpu_rdy_ndtack low to cpu_a[14:0] and cpu_a_das invalid | 0 | | | ns |
| t4 | cpu_rdy_ndtack high -- $\overline{\text{cpu_cs}}$ and $\overline{\text{cpu_rd_ds}}$ asserted to cpu_rdy_ndtack driving one | 0 | | 12 | ns |
| t5 | cpu_rdy_ndtack delay -- $\overline{\text{cpu_cs}}$ and $\overline{\text{cpu_rd_ds}}$ asserted to cpu_rdy_ndtack asserted | | | See Table 205, “t5 Read Access Times,” on page 177. | ns |
| t6 | cpu_rdy_ndtack hold -- $\overline{\text{cpu_cs}}$ or $\overline{\text{cpu_rd_ds}}$ de-asserted to cpu_rdy_ndtack driving one | 0 | | 10 | ns |
| t7 | cpu_rdy_ndtack high impedance -- cpu_rdy_ndtack driving one to cpu_rdy_ndtack high-impedance | 2 | | 8 | ns |
| t9 | Data to cpu_rdy_ndtack delay -- cpu_d[15:0] valid to cpu_rdy_ndtack asserted | upclk - 4 | | | ns |
| t10 | Data output hold -- $\overline{\text{cpu_cs}}$ or $\overline{\text{cpu_rd_ds}}$ de-asserted to cpu_d[15:0] invalid | 0 | | 10 | ns |
| t11 | cpu_r/$\overline{\text{w}}$ high to both $\overline{\text{cpu_cs}}$ and $\overline{\text{cpu_ds}}$ asserted | 0 | | | ns |
| t12 | $\overline{\text{cpu_cs}}$ or $\overline{\text{cpu_ds}}$ high to cpu_r/$\overline{\text{w}}$ low | 0 | | | ns |

**Table 204 - Non-Multiplexed CPU Interface - Motorola Mode**

| Symbol | Description | Burst Length (read only) | Max | Unit |
|---|---|---|---|---|
| t5 | Register and internal memory access | 1-word | 740 | ns |
| t5 | SSRAM | 1-word | 1.07 | µs |
| t5 | SSRAM | 8-words | 1.44 | µs |
| t5 | SSRAM | 128-words | 8.78 | µs |
| t5 | SDRAM | 1-word | 1.16 | µs |
| t5 | SDRAM | 8-words | 2.32 | µs |
| t5 | SDRAM | 128-words | 24.9 | µs |

**Table 205 - t5 Read Access Times**

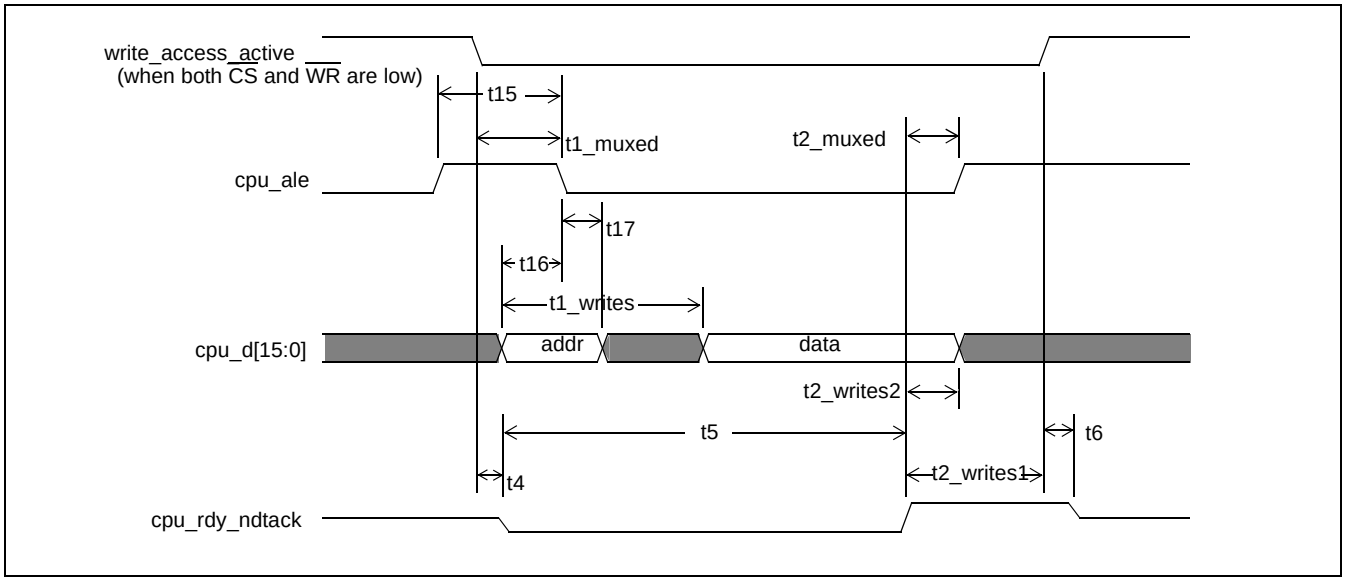



**Figure 75 - Multiplexed CPU Interface - Intel Mode - Write Access**

| Symbol | Description | Min | Typical | Max | Unit |
|---|---|---|---|---|---|
| t1_muxed | write_access_active falling edge to cpu_ale fall | | | 2 * upclk - 4 | ns |
| t1_writes | write_access_active falling edge to cpu_d valid | | | 2 * upclk - 4 | ns |
| t2_muxed | cpu_rdy_ndtack rising edge to cpu_ale rising edge | 0 | | | ns |
| t2_writes1 | cpu_rdy_ndtack rising edge to write_access_active rising edge | 0 | | | ns |
| t2_writes2 | cpu_rdy_ndtack rising edge to cpu_d invalid | 0 | | | ns |
| t4 | write_access_active falling edge to cpu_rdy_ndtack falling edge | 0 | | 12 | ns |
| t5 | Write Access Time | | | 740 | ns |
| t6 | write_access_active rising edge to cpu_rdy_ndtack tri-state | 0 | | 10 | ns |
| t15 | cpu_ale high pulse width | 5 | | | ns |
| t16 | cpu_d valid to cpu_ale falling edge | 5 | | | ns |
| t17 | cpu_ale falling edge to cpu_d invalid | 5 | | | ns |

**Table 206 - Multiplexed CPU Interface - Intel Mode - Write Access**

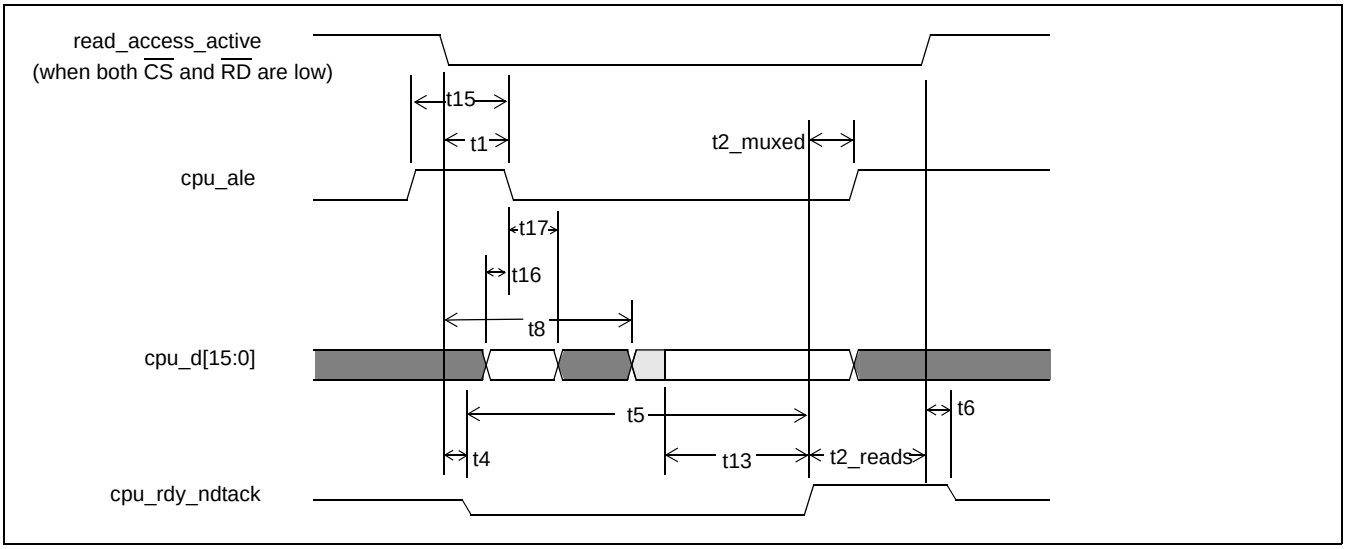



**Figure 76 - Multiplexed CPU Interface - Intel Mode - Read Access**

| Symbol | Description | Min | Typical | Max | Unit |
|---|---|---|---|---|---|
| t1 | read_access_active falling edge to cpu_ale fall | | | 2 * upclk - 4 | ns |
| t2_muxed | cpu_rdy_ndtack rising edge to cpu_ale rising edge | 0 | | | ns |
| t2_reads | cpu_rdy_ndtack rising edge to read_access_active rising edge | 0 | | | ns |
| t4 | read_access_active falling edge to cpu_rdy_ndtack falling edge | 0 | | 12 | ns |
| t5 | Read Access Time | | | see Table 205 | |

**Table 207 - Multiplexed CPU Interface - Intel Mode - Read Access**

| | | | | | |
|---|---|---|---|---|---|
| t6 | read_access_active rising edge to cpu_rdy_ndtack tri-state | 0 | | 10 | ns |
| t8 | read_access_active falling edge to cpu_d driven | 3 * upclk - 4 | | | ns |
| t13 | cpu_d valid to cpu_rdy_ndtack rising edge | upclk - 4 | | | ns |
| t15 | cpu_ale high pulse width | 5 | | | ns |
| t16 | cpu_d valid to cpu_ale falling edge | 5 | | | ns |
| t17 | cpu_ale falling edge to cpu_d invalid | 5 | | | ns |

**Table 207 - Multiplexed CPU Interface - Intel Mode - Read Access (continued)**

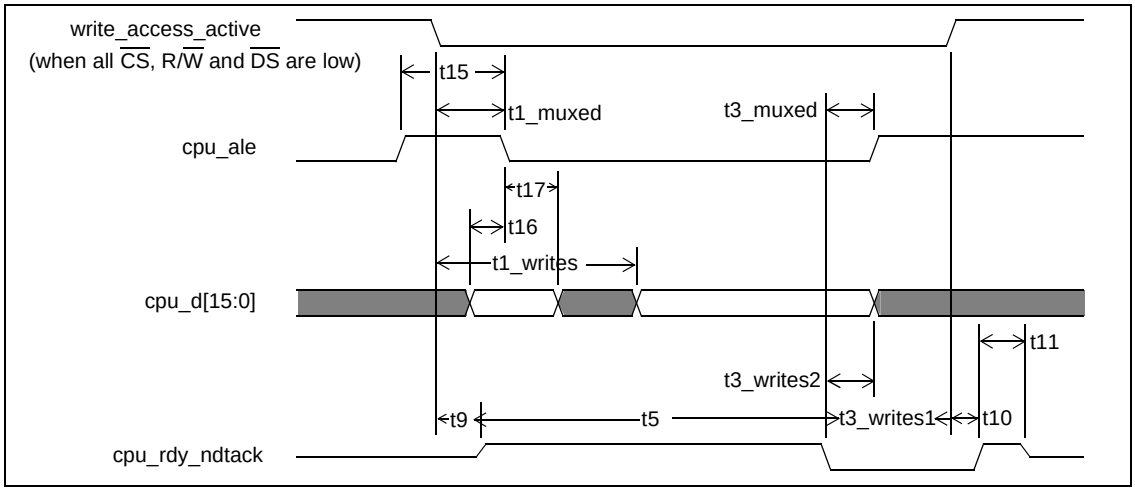



**Figure 77 - Multiplexed CPU Interface - Motorola Mode - Write Access**

| Symbol | Description | Min | Typical | Max | Unit |
|---|---|---|---|---|---|
| t1_muxed | write_access_active falling edge to cpu_ale fall | | | 2 * upclk - 4 | ns |
| t1_writes | write_access_active falling edge to cpu_d valid | | | 2 * upclk - 4 | ns |
| t3_muxed | cpu_rdy_ndtack falling edge to cpu_ale rising edge | 0 | | | ns |
| t3_writes1 | cpu_rdy_ndtack falling edge to write_access_active rising edge | 0 | | | ns |
| t3_writes2 | cpu_rdy_ndtack falling edge to cpu_d invalid | 0 | | | ns |
| t5 | Write Access Time | | | 740 | ns |
| t9 | write_access_active falling edge to cpu_rdy_ndtack driven high | 0 | | 12 | ns |
| t10 | write_access_active rising edge to cpu_rdy_ndtack rising edge | 0 | | 10 | ns |
| t11 | cpu_rdy_ndtack rising edge to cpu_rdy_ndtack tri-state | 2 | | 8 | ns |
| t15 | cpu_ale high pulse width | 5 | | | ns |
| t16 | cpu_d valid to cpu_ale falling edge | 5 | | | ns |
| t17 | cpu_ale falling edge to cpu_d invalid | 5 | | | ns |

**Table 208 - Multiplexed CPU Interface - Motorola Mode - Write Access**

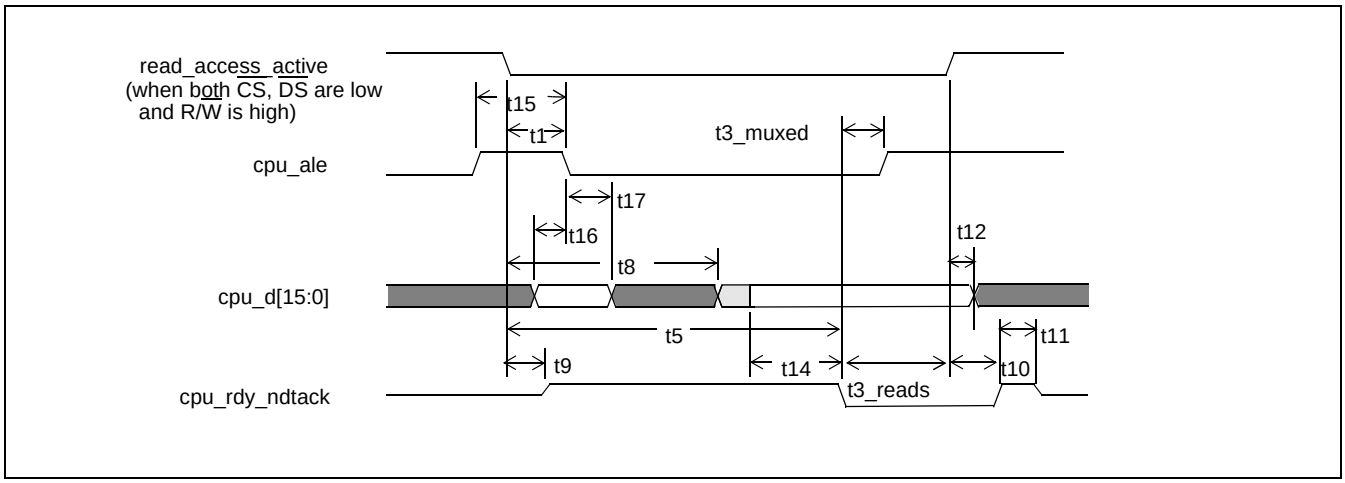



**Figure 78 - Multiplexed CPU Interface - Motorola Mode - Read Access**

| Symbol | Description | Min | Typical | Max | Unit |
|---|---|---|---|---|---|
| t1 | read_access_active falling edge to cpu_ale fall | | | 2 * upclk - 4 | ns |
| t3_muxed | cpu_rdy_ndtack falling edge to cpu_ale rising edge | 0 | | | ns |
| t3_reads | cpu_rdy_ndtack falling edge to read_access_active rising edge | 0 | | | ns |
| t5 | Read Access Time | | | see Table 205 | |
| t8 | read_access_active falling edge to cpu_d driven | 3 * upclk - 4 | | | ns |
| t9 | read_access_active falling edge to cpu_rdy_ndtack driven high | 0 | | 12 | ns |
| t10 | read_access_active rising edge to cpu_rdy_ndtack rising edge | 0 | | 10 | ns |
| t11 | cpu_rdy_ndtack rising edge to cpu_rdy_ndtack tri-state | 2 | | 8 | ns |
| t15 | cpu_ale high pulse width | 5 | | | ns |
| t16 | cpu_d valid to cpu_ale falling edge | 5 | | | ns |
| t17 | cpu_ale falling edge to cpu_d invalid | 5 | | | ns |

**Table 209 - Multiplexed CPU Interface - Motorola Mode - Read Access**

### 4.3.1 UTOPIA Interface

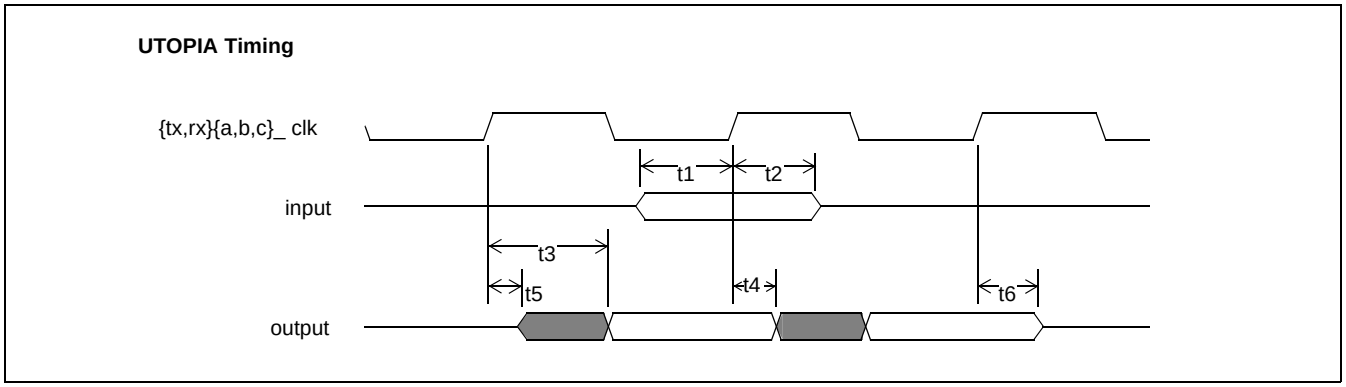



**Figure 79 - UTOPIA Timing**

| Symbol | Characteristics | Min | Type | Max | Units |
|---|---|---|---|---|---|
| t1 | Input setup time | 4 | | | ns |
| t2 | Input hold time | 1 | | | ns |
| t3 | Clock to data valid | | | 12 | ns |
| t4 | Clock to data change | 2 | | | ns |
| t5 | Clock rising to signal driven | 1 | | | ns |
| t6 | Clock rising to signal tri-state | 1 | | 20 | ns |

**Table 210 - UTOPIA Timing**

### 4.3.2 External Memory Interface

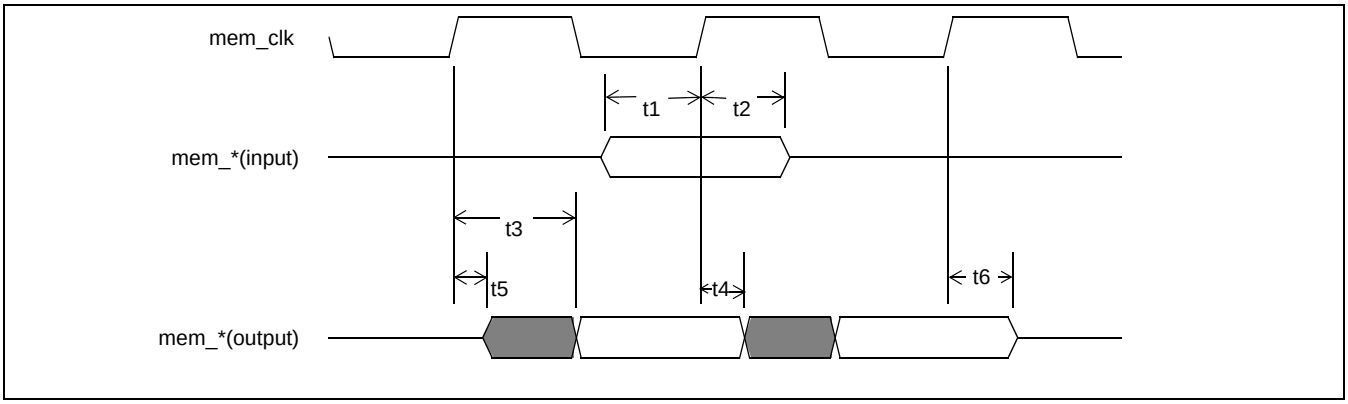



**Figure 80 - External Memory Timing (both SSRAM and SDRAM)**

| Symbol | Characteristics | Min | Type | Max | Units | Load |
|---|---|---|---|---|---|---|
| t1 | Input setup time | 2 | | | ns | |
| t2 | Input hold time | 0 | | | ns | |
| t3 | Clock to data valid | | | 8.30 | ns | 50 pF |
| t4 | Clock to data change | 2 | | | ns | 50 pF |
| t5 | Clock rising to signal driven | 2 | | | ns | |
| t6 | Clock rising to signal tri-state | | | 10 | ns | |

**Table 211 - External Memory Timing**

### 4.3.3 H.100/H.110 Interface

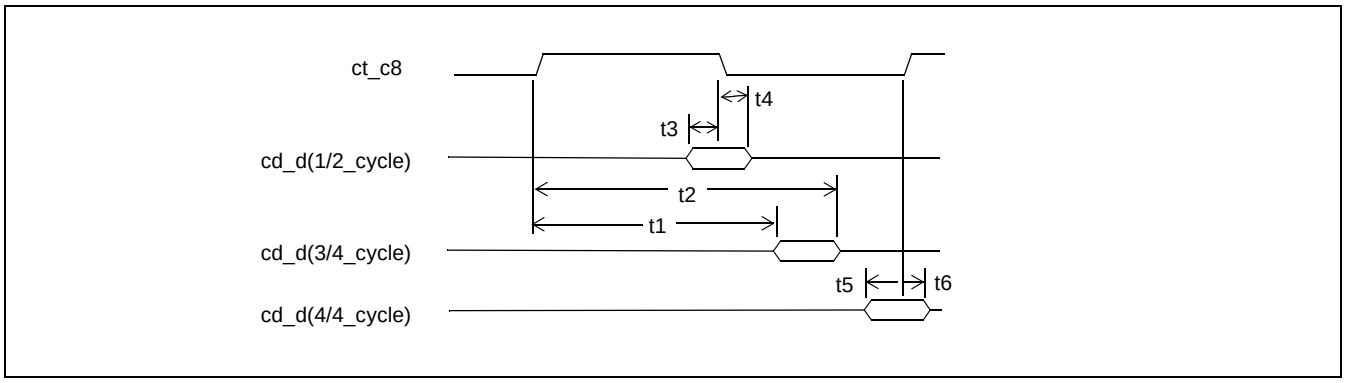



**Figure 81 - H.100/H.110 Timing - H.100 Input Sampling**

| Symbol | Description | Min | Typical | Max | Unit |
|---|---|---|---|---|---|
| t1 | ct_c8 rise to ct_d valid | | | 69 | ns |
| t2 | ct_c8 rise to ct_d invalid | 102 | | | ns |
| t3 | ct_d valid to ct_c8 fall | 3 | | | ns |
| t4 | ct_c8 fall to ct_d invalid | 1 | | | ns |
| t5 | ct_d valid to ct_c8 rise | 5 | | | ns |
| t6 | ct_c8 rise to ct_d invalid | 0 | | | ns |

**Table 212 - H.100/H.110 Timing - H.100 Input Sampling**

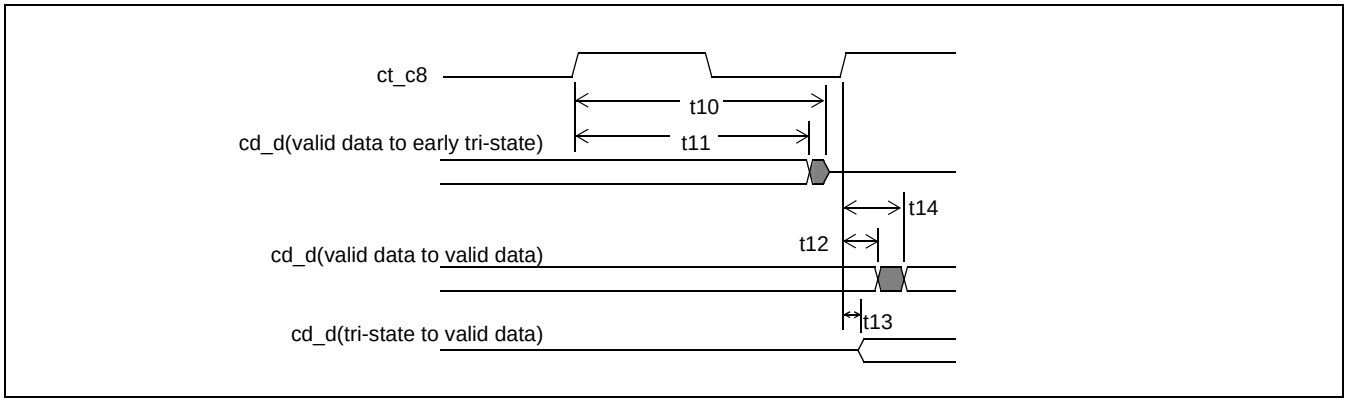



**Figure 82 - H.100/H.110 Timing - H.100 Output**

| Symbol | Description | Min | Typical | Max | Unit | Load |
|---|---|---|---|---|---|---|
| t10 | ct_c8 rise to ct_d tri-state | | | 122 | ns | 200 pf |
| t11 | ct_c8 rise to ct_d invalid | 102 | | | ns | 200 pf |
| t12 | ct_c8 rise to ct_d invalid | 2 | | | ns | 200 pf |
| t13 | ct_c8 rise to ct_d driven | 2 | | | ns | 200 pf |
| t14 | ct_c8 rise to ct_d valid | | | 22 | ns | 200 pf |

**Table 213 - H.100/H.110 Timing - H.100 Output**

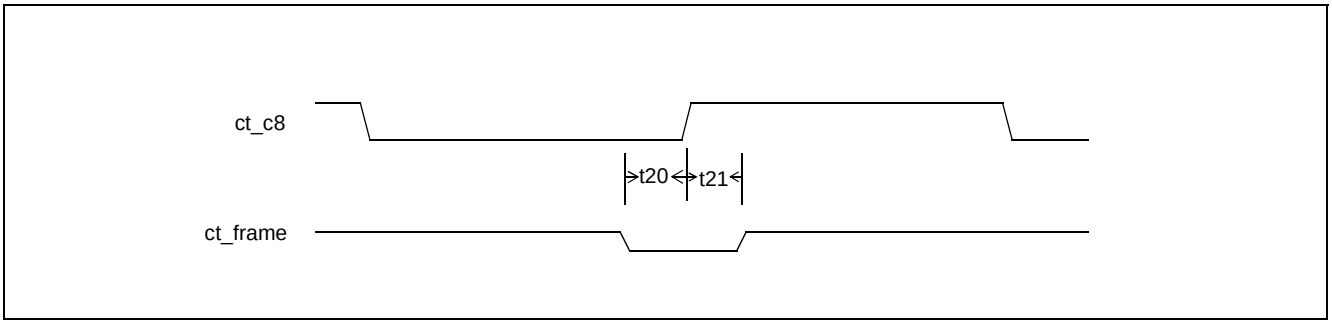



**Figure 83 - H.100/H.110 Timing - H.100 Frame Sampling**

| Symbol | Description | Min | Typical | Max | Unit |
|---|---|---|---|---|---|
| t20 | ct_frame valid to ct_c8 rise | 5 | | | ns |
| t21 | ct_c8 rise to ct_frame invalid | 5 | | | ns |

**Table 214 - H.100/H.110 Timing - H.100 Frame Sampling**

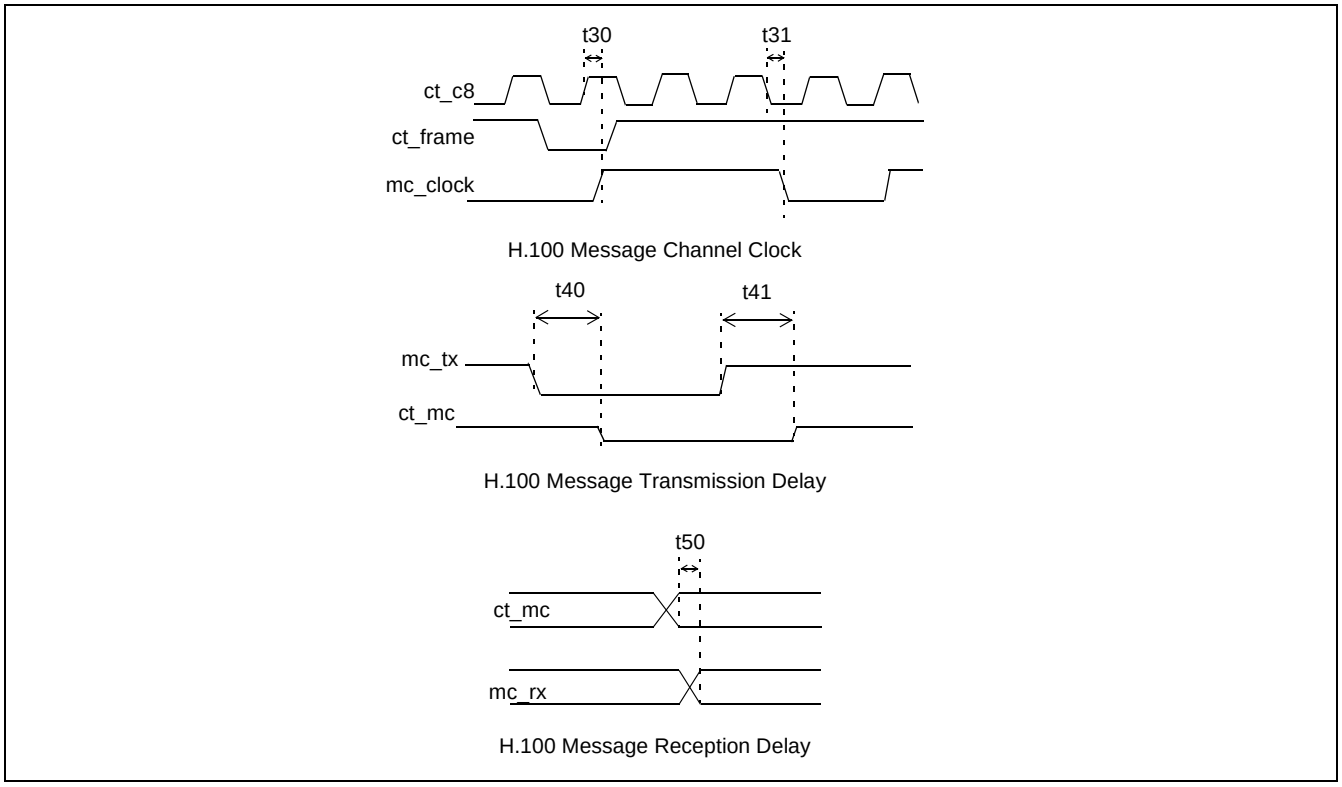



**Figure 84 - H.100/H.110 Message Timing**

| Symbol | Description | Min | Typical | Max | Unit | Load |
|---|---|---|---|---|---|---|
| t30 | ct_c8 rise to mc_clock rise | | | 15 | ns | |
| t31 | ct_c8 fall to mc_clock fall | | | 15 | ns | |
| t40 | mc_tx fall to ct_mc low | 3 | | 15 | ns | 200 pf |
| t41 | mc_tx rise to ct_mc tri-state | 3 | | 15 | ns | 200 pf |
| t50 | ct_mc fall to mc_rx fall | 3 | | 15 | ns | |

**Table 215 - H.100/H.110 Message Timing**

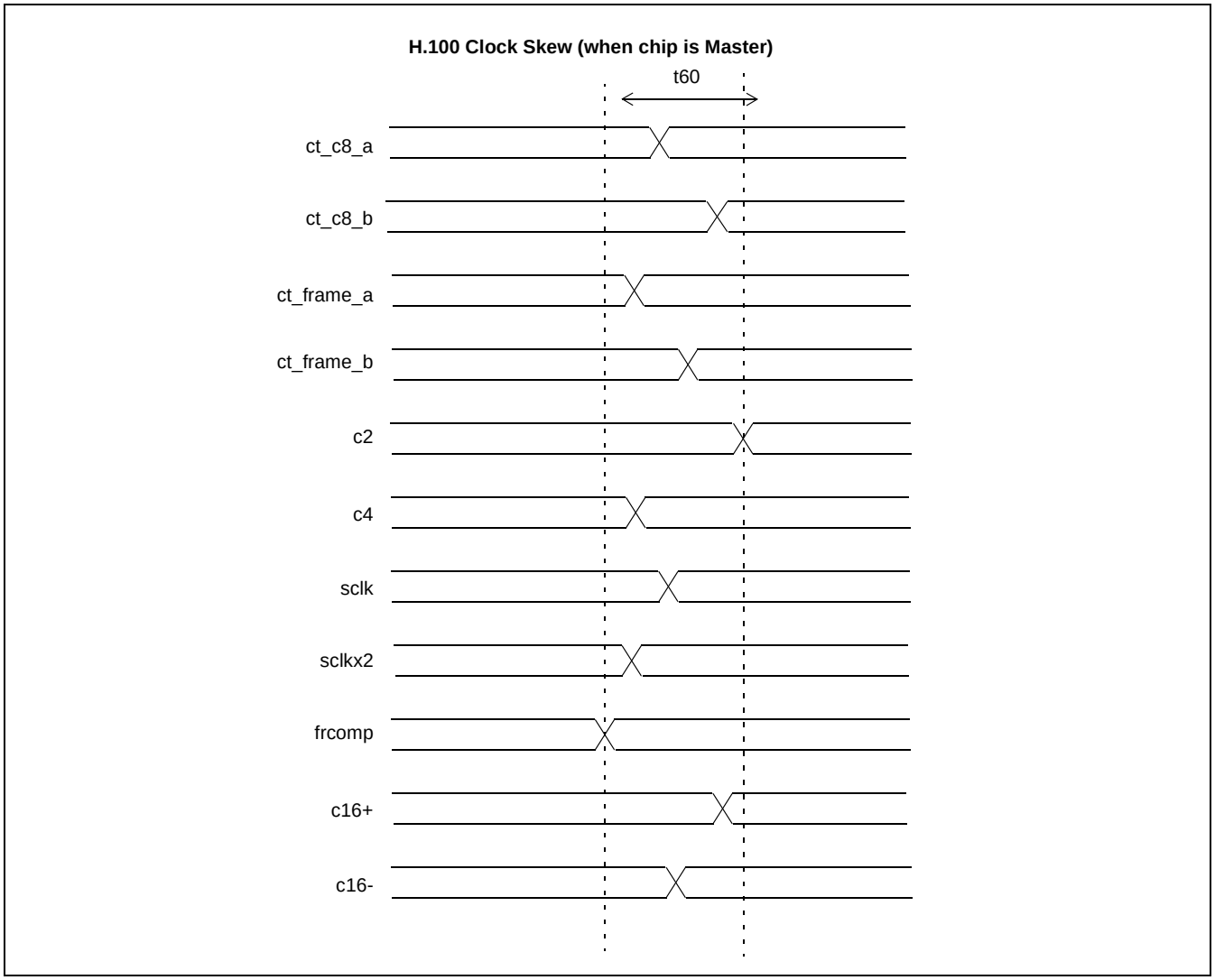



**Figure 85 - H.100/H.110 Clock Skew**

| Symbol | Description | Min | Typical | Max | Unit | Load |
|---|---|---|---|---|---|---|
| t60 | maximum skew when signals generated by MT90502 | | | 5 | ns | 200 pF |

**Table 216 - H.100/H.110 Clock Skew Table**

# 5.0 Glossary of Terminology

## 5.1 Standard Terms and Abbreviations

**AAL0: A**TM **A**daptation **L**ayer **0**. AAL0 is a straight packaging of 48 bytes of data within an ATM cell. AAL0 can be used to treat either data cells (managed by CPU) or CBR cells (managed by TX SAR and RX SAR).

**AAL2: A**TM **A**daptation **L**ayer **2**. AAL2 is used to transport constant bit rate (CBR) and variable bit rate (VBR) data on ATM. AAL2 cells are made up of AAL2 CPS-Packets of different lengths containing different kinds of data.

**ADPCM: A**daptive **D**ifferential **P**ulse **C**ode **M**odulation. ADPCM is a compression standard that allows the encoding of PCM data at rates of 40, 32, 24 and 16 kbps.

**CBR: C**onstant **B**it **R**ate. CBR data means that cells on that particular channel are sent out at a regular rate. CBR is applicable to voice channels.

**CRC: C**yclic **R**edundancy **C**heck. The CRC is a method of error detection and correction that is applied to a certain field of data. CRC is an efficient method of error detection because the odds of erroneously detecting a correct payload are low.

**FIFO: F**irst **I**n, **F**irst **O**ut. A FIFO memory is one in which the first byte to have been written into the memory is the first one to be read from the read port.

**GFC: G**eneric **F**low **C**ontrol. The GFC field is kept in the 4 highest bits of an ATM cell's header and is used for local functions (not carried end-to-end). The default value is "0000", meaning that GFC protocol is not enforced.

**H.100/H.110:** A TDM bus standard developed by ECTF to provide backward compatibility to existing TDM busses with more bandwidth and potential for development.

**HDLC: H**igh-level **D**ata **L**ink **C**ontrol. An encapsulation protocol that defines specific bit patterns as de-limiters and thus allows transmission of data over a serial link. In the MT90502, HDLC is used to carry variable-length packets on the H.100/H.110 bus.

**OC-3: O**ptical **C**arrier level-**3**. A SONET channel that carries a bandwidth of 155.55 Mbps.

**PCM: P**ulse **C**ode **M**odulation. PCM is the basic method of encoding an analog voice signal into digital form.

**PHY: PHY**sical layer. The bottom layer of the ATM Reference Model, it provides ATM cell transmission over the physical interfaces that interconnect the various ATM devices.

**PLL: P**hase **L**ock **L**oop. A phase lock loop is a component that generates an output clock by synchronizing itself to an input clock. PLLs are often used to multiply the frequency of clocks.

**RAM: R**andom **A**ccess **M**emory. RAM is the main memory in the computer. It is called "random" because any random address can be accessed in an equal amount of time.

**SAR: S**egmentation **A**nd **R**eassembly. Method of partitioning, at the source, frames into ATM cells and reassembling, at the destination, these cells back into information frames. SAR is the lower sub-layer of the AAL which inserts data from the information frames into cells and then add the required header, trailer, and/or padding bytes to create 48-byte payloads to be transmitted to the ATM layer.

**TDM: T**ime **D**ivision **M**ultiplexing. TDM busses carry voice data divided according to frames. In a single 125 us frame, the TDM bus will have carried one byte from each channel it contains.

**UTOPIA: U**niversal **T**est and **O**perations **P**HY **I**nterface for **A**TM. A PHY-level interface to provide connectivity between ATM components.

**VC: V**irtual **C**ircuit. VCs define a point-to-point connection between two nodes in a network. A single ATM cell carries data that belongs to a single VC.

**VCI: V**irtual **C**hannel **I**dentifier. This is the label given to an ATM VC to identify it and determine its destination. The VCI is a 16-bit number that is included in the header of an ATM cell.

**VPI: V**irtual **P**ath **I**dentifier. A virtual path determines the way an ATM cell should be routed. The VPI is an 8-bit (in UNI) or 12-bit (in NNI) number that is included in the header of an ATM cell.

### 5.2 Terms specific to AAL2

**CID: C**hannel **ID**entifier. The CID is an 8-bit field that identifies an AAL2 CPS-Packet and determines which of the 255 AAL2 channels on this VC it belongs to. The value of 00h is illegal for the CID, values 01h to 07h are denoted as reserved for use by the AAL type 2.

**EDU**: **E**ncoding **D**ata **U**nit. A group of 8 PCM bytes or 8 ADPCM samples that represent 1 ms of voice traffic. The size of every PCM or ADPCM CPS-Packet sent on AAL2 is an integer number of EDUs.

**LI: L**ength **I**ndicator. The LI is a 6-bit field encoded with a value that is one less than the number of octets in the CPS-Packet Payload portion of a CPS-Packet.

**SID: S**ilent **I**nsertion **D**escriptor. A SID CPS-Packet is an AAL2 CPS-Packet containing a single byte of payload that is inserted when a valid CPS-Packet has been suppressed because it was silent. The payload byte indicates the energy level of the voice that was suppressed.

**UUI**: **U**ser-to-**U**ser **I**ndication. The UUI is a 5-bit field contained within the AAL2 header that is used to indicate the type of an AAL2 CPS-Packet. When indicating voice data, the UUI is often used as a 4-bit sequence number.

### 5.3 Terms specific to this specification

**AAL2 Channel:** Any sub-channel carried by an AAL2 VC. An AAL2 channel is uniquely identified by its CID and the VPI/VCI of the VC it belongs to.

**AAL2 CPS-Packet:** A CPS-Packet contained within one or many AAL2 cells. AAL2 CPS-Packets contain 3 bytes of overhead (including CID, LI, UUI and HEC) and from 1 to 64 bytes of payload. Because of this 64-byte maximum, they can straddle many cells. Also known as CPS-packet.

**HDLC Stream:** A group of HDLC channels that are carried over the same time slots. HDLC CPS-Packets in streams have an address byte that indicates to which HDLC channel they belong. Usually, HDLC streams carry a series of channels communicating to and from the same agent (e.g. a DSP). HDLC Streams must be carried over a single H.100 stream and over one or multiple consecutive time slots on that stream.

**HDLC channel:** An HDLC channel carries CPS-Packets destined to the same AAL2 channel. All CPS-Packets of an HDLC channel are carried by the same HDLC stream.

**PDV: P**acket **D**elay **V**ariation. AAL2 CPS-Packets arrive with a certain delay with respect to when they were sent. PDV is a measure of how much that delay varies on an AAL2 channel. PDV measures the peak-to-peak packet delay throughout the network. PDV is only relevant on CBR connections.

**Time Slot:** In this document, the term time slot is often used to define a combination of a time slot and a stream on the H.100 bus. Thus a time slot would represent a single 8-bit slot every 125 us on the TDM bus.

### 5.4 Register types

**CNT**: **C**ounter. Events in the MT90502 will cause the counter to increment.

**CRL**: **C**ounter **R**oll-**O**ver: This bit indicates its respect counter has wrapped.

**IE**: **I**nterrupt **E**nable. This is a register bit that enables a status event to generate an interrupt. This bit is always active-high.

**PC**: **P**rocess **C**ontrol bit. This is a register bit type that is written to ‘1’ to initiate a hardware process. When the process completes, the hardware clears the bit.

**PUL**: **P**ulse. This bit is used to set an event. Setting this bit, creates a pulse in the MT90502 of 1 clock period. The hardware then clears this bit.

**RO: R**ead **O**nly. This serves to define registers that cannot be written to by the CPU.

**ROL: R**ead **O**nly **L**atch. This defines status bits. Status bits cannot be written to ‘1’ by the CPU; however, once the status bit is set, the CPU can clear it by writing a ‘1’ over it.

**RW: R**ead **W**rite. This type of register bit will be readable and writable by the CPU.

**TS**: **T**est **S**tatus. This type is for test purposes only and should not be written by the user.

**WO**: **W**rite **O**nly. This type of register bit is writable by the CPU. The value read back by the CPU may not reflect the true value of the bit.

### 5.5 Units and Conventions

All numbers in this document are decimal unless otherwise specified.

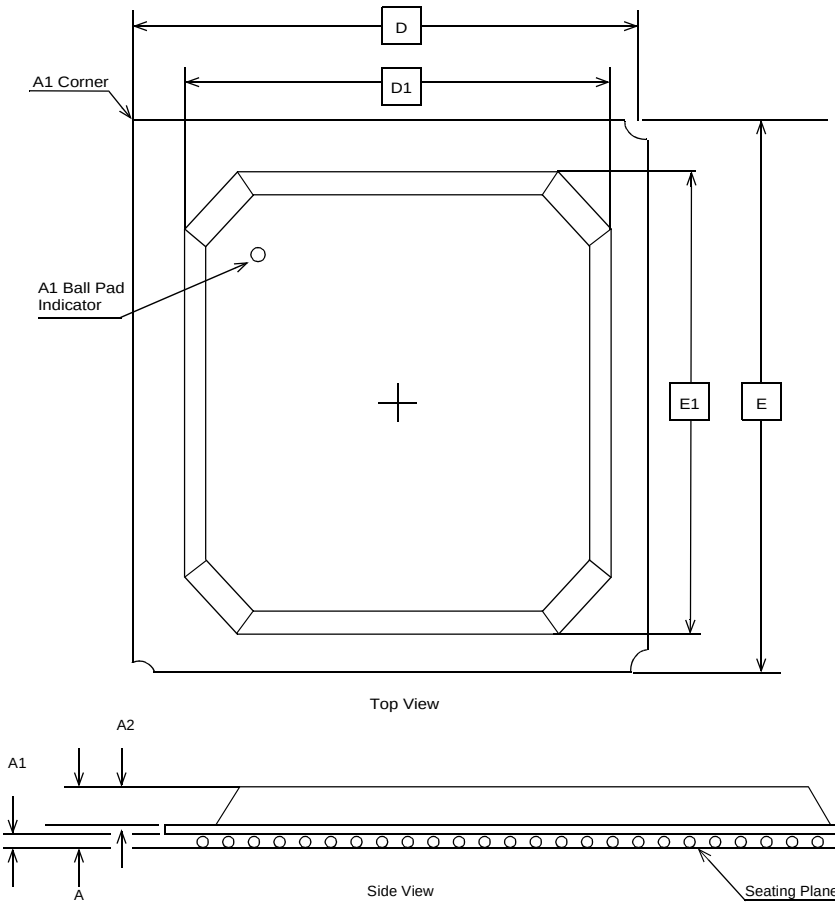
Hexadecimal numbers can be identified by the ‘h’ suffix (e.g. 00A5h).

Binary numbers are either in double quotes for multiple bits or in single quotes for individual bits (e.g. “1001”, ‘0’).

All addresses are specified in hexadecimal and point to bytes.

Addresses are converted from bytes to words to double words using the little endian format, unless otherwise specified.

## 6.0 Mechanical Drawing




| Dimension | Min | Max |
|---|---|---|
| A | 1.92 (2.12) | 2.32 (2.54) |
| A1 | 0.50 | 0.70 |
| A2 | 1.12 | 1.22 |
| D | 35.00 BSC | |
| D1 | 30.00 | 30.70 |
| E | 35.00 BSC | |
| E1 | 30.00 | 30.70 |
| I | 1.625 REF | |
| J | 1.625 REF | |
| b | 0.60 | 0.90 |
| e | 1.27 BSC | |
| N | 456 | |
| 2 Layers (4 Layers) | | |

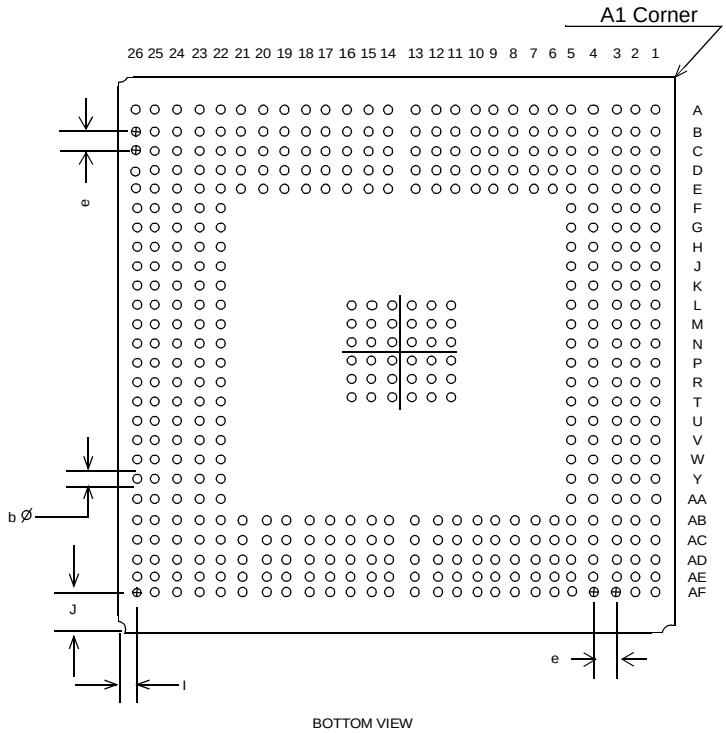



**Figure 86 - MT90502 Package Outline Drawing (456 PBGA)**



**For more information about all Zarlink products**
**visit our Web Site at**

## www.zarlink.com

Information relating to products and services furnished herein by Zarlink Semiconductor Inc. or its subsidiaries (collectively "Zarlink") is believed to be reliable. However, Zarlink assumes no liability for errors that may appear in this publication, or for liability otherwise arising from the application or use of any such information, product or service or for any infringement of patents or other intellectual property rights owned by third parties which may result from such application or use. Neither the supply of such information or purchase of product or service conveys any license, either express or implied, under patents or other intellectual property rights owned by Zarlink or licensed from third parties by Zarlink, whatsoever. Purchasers of products are also hereby notified that the use of product in certain ways or in combination with Zarlink, or non-Zarlink furnished goods or services may infringe patents or other intellectual property rights owned by Zarlink.

This publication is issued to provide information only and (unless agreed by Zarlink in writing) may not be used, applied or reproduced for any purpose nor form part of any order or contract nor to be regarded as a representation relating to the products or services concerned. The products, their specifications, services and other information appearing in this publication are subject to change by Zarlink without notice. No warranty or guarantee express or implied is made regarding the capability, performance or suitability of any product or service. Information concerning possible methods of use is provided as a guide only and does not constitute any guarantee that such methods of use will be satisfactory in a specific piece of equipment. It is the user's responsibility to fully determine the performance and suitability of any equipment using such information and to ensure that any publication or data used is up to date and has not been superseded. Manufacturing does not necessarily include testing of all functions or parameters. These products are not suitable for use in any medical products whose failure to perform may result in significant injury or death to the user. All products and materials are sold and services provided subject to Zarlink's conditions of sale which are available on request.

Purchase of Zarlink's $I^2C$ components conveys a licence under the Philips $I^2C$ Patent rights to use these components in and $I^2C$ System, provided that the system conforms to the $I^2C$ Standard Specification as defined by Philips.

Zarlink, ZL and the Zarlink Semiconductor logo are trademarks of Zarlink Semiconductor Inc.

Copyright Zarlink Semiconductor Inc. All Rights Reserved.

TECHNICAL DOCUMENTATION - NOT FOR RESALE