Atmel ATPL00B

FSK Power Line Communications SoC

DATASHEET

Features

- Core
  - ADD8051C3A enhanced 8051 core
  - Speedups up to x5 vs. standard 8051 microcontroller
- 128Kbytes internal SRAM
- In-circuit serial flash programming
- Auto boot-loading program from serial flash
- Media Access Control
  - Convolutional and block (FEC) channel coding, Viterbi decoding
  - Hardware CRC error detection and FEC error correction
  - By-pass mode to support earlier no-MAC FSK modem software
- Modem
  - Power Line Carrier Modem for 50 and 60 Hz mains
  - Carrier Frequency: 132.5KHz
  - Baud rate Selectable: 600 to 4800 bps
  - Half Duplex communication
  - Receiver Sensitivity: Up to 44dBµVrms
- Peripherals
  - Three 2-wire UARTs
  - Two SPI. SPI to serial flash and External RTC. Buffered SPI to external metering IC
  - Programmable Watchdog
  - Quad dimmer in/out
  - Up to 20 I/O lines
- Package
  - 144-lead LQFP, 16 x 16 mm, pitch 0.4 mm
  - Pb-free and RoHS compliant
- Typical Applications
  - Automated Meter Reading (AMR) & Advanced Meter Management (AMM)
  - Street lighting
  - Home Automation

Description

The ATPL00B is a Power Line Communications System on Chip that implements a full PLC node using FSK modulations and includes a hardwired Medium Access Controller (ADD1210). It has been developed to reduce the CPU computational load in PLC systems. Thus, the microcontroller is free to be used in the applications tasks.

MAC functional capabilities of ATPL00B (performed in ADD1210 Medium Access Controller) involve the construction of message packets, adding convolutional or FEC (Forward Error Correction) codes to bytes and FCS (Frame Check Sequence) to packets. In reception, the MAC provides frame detection and Viterbi decoding or FCS and FEC correction.

ATPL00B MAC design is versatile and allows users to create a wide range of datagram structures. The MAC shall be set in a bypass mode allowing direct connection between the microcontroller and the modem to support old FSK software that doesn't include the MAC.

ATPL00B PLC modem (ADD1310) is based on a Frequency-shift keying (FSK) Modulation Scheme supporting Minimum Shift Frequency (MSK) in the C-Band with carrier frequency of 132,5KHz. It shall work using a single power supply of 3.3V and a few external components, supporting several Analog Front End (AFE) configurations suitable for Home Automation purposes. It shall replace the traditional analog PLC modem and can use the same software libraries or a simplified version if the hardwired MAC is used. The PLC modem fits CENELEC C-band and EN50065-1 access rules, and has receiver sensitivity up to 44dBµVrms (158 µVrms).

ATPL00B core (ADD8051C3A) includes all features of the standard 8051, with an average speed up to x5 and some additional features.

The microcontroller includes some specific peripherals as 4 input / 4 output dimmers for power regulation (phase angle control), and also capable of generating Pulse-Width Modulation (PWM) control.

A flash program loader allows downloading the microcontroller program in a standard SPI serial flash memory and executing it from internal SRAM. In the start-up process, the program is uploaded from serial flash to the internal 128Kbytes of SRAM before starting the execution. After that, the free space in the serial flash shall be used to store application data. ATPL00B includes an encryption engine for code protection. Using a larger flash, several programs may be stored at the same time and the microcontroller shall switch from one program to another. This feature could be used to reprogram the SoC using PLC downloading.

Table of Contents

1. Block Diagram 6

2. Package and Pinout 7
   - 2.1 144-Lead LQFP Package Outline 7
   - 2.2 144-Lead LQFP Pinout 8

3. Pin Description 10

4. Processor and Architecture 17
   - 4.1 ADD8051C3A Microcontroller Description 17
   - 4.2 Core Pinout Description 18
   - 4.3 Memory Organization 19
     - 4.3.1 Program Memory 20
     - 4.3.2 Extended Addressing 21
     - 4.3.3 Data Memory 25
     - 4.3.4 SFR Registers 26
   - 4.4 Instruction Set 29
     - 4.4.1 Program Status Word 29
     - 4.4.2 Addressing Modes 29
     - 4.4.3 Arithmetic Instructions 30
     - 4.4.4 Logical Instructions 31
     - 4.4.5 Data Transfer Instructions 33
     - 4.4.6 Boolean Instructions 34
     - 4.4.7 Jump Instructions 35
   - 4.5 CPU Timing 37
     - 4.5.1 Reset 38
     - 4.5.2 Power Saving Modes 40
     - 4.5.3 Idle Mode 40
     - 4.5.4 Power-Down Mode 40
   - 4.6 Interrupts 41
     - 4.6.1 Interrupt Enabling 41
     - 4.6.2 Interrupt Priorities 41
     - 4.6.3 Interrupt Handling 44
   - 4.7 I/O Ports 45
     - 4.7.1 I/O Configurations 46
     - 4.7.2 Read-Modify-Write Feature 47
     - 4.7.3 Accessing External Memories 47
   - 4.8 Debug Mode 48

5. Timers 49
   - 5.1 Timer 0 and Timer 1 49
     - 5.1.1 Timer Mode 0 50
     - 5.1.2 Timer Mode 1 51
     - 5.1.3 Timer Mode 2 51
     - 5.1.4 Timer Mode 3 51
   - 5.2 Timer 2 52
     - 5.2.2 Capture mode 53
     - 5.2.3 Auto-Reload mode 53
     - 5.2.4 Clock-Out mode 54
     - 5.2.5 Baud rate Generator mode 55
   - 5.3 Watchdog (timer 3) 56

6. Standard Serial Interfaces 58
   - 6.1 Serial Port modes 58
     - 6.1.1 Mode 0 (shift register mode) 58
     - 6.1.2 Mode 1 (8-bit UART) 59
     - 6.1.3 Modes 2 and 3 (9-bit UART) 59
   - 6.2 Serial Port Timers (Timers 11, 12 and 14) 60

7. Serial Peripheral Interfaces SPI0 and SPI1 64
7.1 SPI description 64
7.1.2 SPI clock phase, polarity and operation 66
7.1.3 SPI0 write collision 68
7.2 SPI Modes 68
7.3 SPI1 buffer operation 70

8. Boot Loader 72
8.1 Pin Description 73
8.2 Flash Programming 74
8.2.1 In-System programming 74
8.2.2 SPI Flash programming 74
8.3 System startup 76
8.3.1 Encrypted firmware requirements 77
8.3.2 Supported Devices 77
8.3.2.1 Serial Number Device 77

9. Dimmer Peripheral 78
9.1 Configuration Registers 79
9.1.1 DIM_CTRL register 80
9.1.2 INP_ST register 81
9.1.3 OUT_ST register 82
9.1.4 OUT_REF registers 83
9.1.5 POLARITY register 84
9.1.6 D_CONF register 85
9.1.7 V_SWC register 86
9.1.8 PWM_PER register 87
9.1.9 INP_SOURCE register 88
9.2 External Circuits 89

10. Media Access Layer 92
10.1 Packet Encapsulation 92
10.2 MAC Architecture 94
10.3 MAC Configuration Registers 94
10.3.1 CTRL register 95
10.3.2 FLAGS1 register 97
10.3.3 FLAGS2 register 99
10.3.4 DATA register 101
10.3.5 TH_PREAM register 102
10.3.6 TH_HEADER register 103
10.3.7 BAUDRATE register 104
10.3.8 TRACK register 105
10.3.9 PREAM register 106
10.3.10 CTRL2 register 107
10.3.11 BC register 108
10.3.12 MODE1 to MODE4 registers 109
10.3.13 VTB_BE_HARD register 110
10.3.14 VTB_BE_SOFT registers 111
10.3.15 FEC_BER register 112
10.4 Operation modes 113
10.4.2 BYPASS mode 113
10.4.3 WHD mode 113
10.4.4 BYTE mode 114
10.4.5 FEC mode 114
10.4.6 VTB mode 114
10.4.7 FCS mode 114
10.4.8 NULL mode 114
10.5 Operation Procedures 114
10.5.1 STOP cycle 114
10.5.2 WHD mode operation procedure 114
10.5.3 Transmission procedure 116

10.5.4 Reception procedure ... 118

11. PLC Modem ... 119

11.1 Frequency coding ... 120
11.2 Modem Transmission and Reception ... 121
11.2.1 Transmission characteristics ... 121
11.2.2 Reception characteristics ... 122
11.3 PLC Modem Configuration registers ... 124
11.3.1 GAIN register ... 125
11.3.2 CONTROL2 register ... 126
11.3.3 CONFIG register ... 127
11.3.4 GFSK_ADDR register ... 128
11.3.5 GFSK_DX registers ... 129
11.3.6 COMP register ... 130
11.3.7 KNX_EN register ... 131
11.3.8 CD_ENABLE register ... 132
11.3.9 CD_DECISION_TIME register ... 133
11.3.10 CD_DECISION_ERROR register ... 134
11.3.11 TXRX_CTL register ... 135
11.3.12 Ri registers ... 136

12. Electrical Characteristics ... 137

12.1 Absolute Maximum Ratings ... 137
12.2 Recommended Operating Conditions ... 138
12.3 DC Characteristics ... 139
12.3.2 V-I curves ... 140
12.4 Power Consumption ... 143
12.5 Thermal Data ... 143
12.6 Oscillator ... 144
12.7 Power on ... 146

13. Mechanical Characteristics ... 147

14. Recommended mounting conditions ... 148

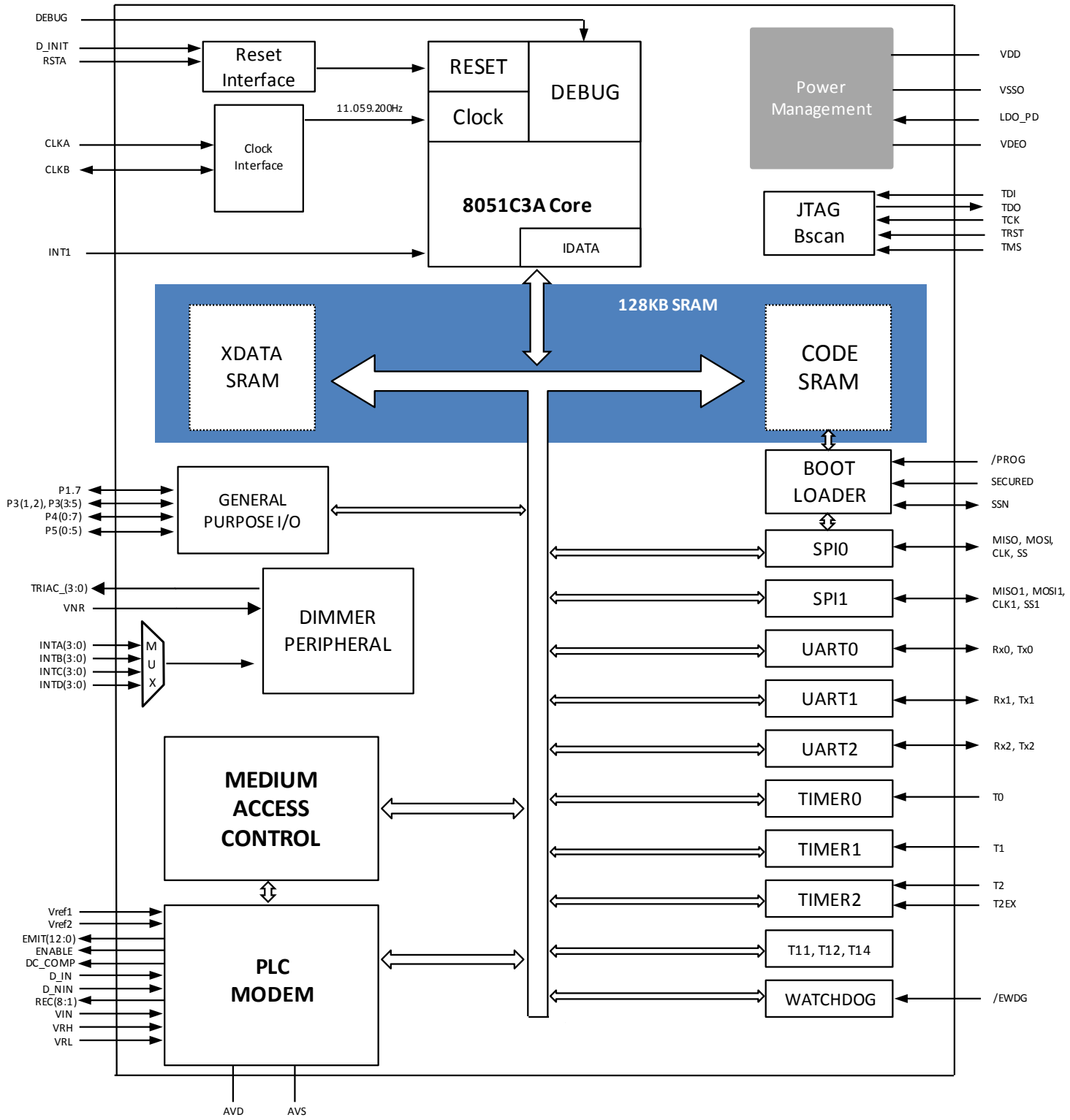
14.1 Conditions of Standard Reflow ... 148
14.2 Manual Soldering ... 149

15. Ordering Information ... 150

16. Revision History ... 151

1. Block Diagram

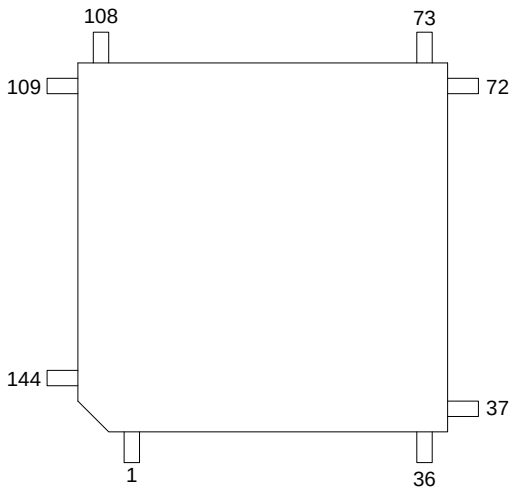
Figure 1-1. ATPL00B 144-pin Block Diagram



2. Package and Pinout

2.1 144-Lead LQFP Package Outline

Figure 2-1. Orientation of the 144-Lead Package



2.2 144-Lead LQFP Pinout

Table 2-1. **ATPL00B 144-Lead LQFP pinout**

PinNo	Pin Name	I/O	I(mA)	Res	HY
1	P3_3/INT1	I/O	±5	PU	-
2	VCC	P	-	-	-
3	GND	P	-	-	-
4	GND	P	-	-	-
5	GND	P	-	-	-
6	TDI	I	±5	PU	-
7	TDO	O	±5	-	-
8	TCK	I	±5	-	-
9	TMS	I	±5	PU	-
10	TRST	I	±5	PU	-
11	D_INIT	I	±5	PD	Y
12	RSTA	I	±5	PD	Y
13	/PROG	I	±5	PU	Y
14	SECURED	I	±5	PD	Y
15	/EWDG	I	±5	PD	Y
16	DEBUG	I	±5	PD	Y
17	VCC	P	-	-	-
18	CLKEB	I/O	-	-	-
19	GND	P	-	-	-
20	CLKEA	I	-	-	-
21	VCC	P	-	-	-
22	GND	P	-	-	-
23	GND	P	-	-	-
24	VDE0	P	-	-	-
25	VDE0	P	-	-	-
26	VSS0	P	-	-	-
27	LDO_PD	I			
28	VDD	P	-	-	-
29	GND	P	-	-	-
30	VCC	P	-	-	-
31	Vref1	I	-	-	-
32	Vref2	I	-	-	-
33	VNR	I	±5	-	Y
34	TRIAC_3	O	±5	-	-
35	TRIAC_2	O	±5	-	-
36	TRIAC_1	O	±5	-	-
37	TRIAC_0	O	±5	-	-
38	P5_5/TXD1/INTA1	I/O	±5	PU	-

PinNo	Pin Name	I/O	I(mA)	Res	HY
39	P5_4/RXD1/INTA0	I/O	±5	PU	-
40	P4_7/T2EX/INTA3	I/O	±5	PU	-
41	P4_6/T2/INTA2	I/O	±5	PU	-
42	P1_7/SSN	I/O	±5	PU	-
43	VCC	P	-	-	-
44	GND	P	-	-	-
45	EMIT_0	O	±X	-	-
46	EMIT_1	O	±X	-	-
47	EMIT_2	O	±X	-	-
48	VCC	P	-	-	-
49	GND	P	-	-	-
50	EMIT_3	O	±X	-	-
51	EMIT_4	O	±X	-	-
52	EMIT_5	O	±X	-	-
53	EMIT_6	O	±X	-	-
54	VCC	P	-	-	-
55	GND	P	-	-	-
56	EMIT_7	O	±X	-	-
57	EMIT_8	O	±X	-	-
58	EMIT_9	O	±X	-	-
59	EMIT_10	O	±X	-	-
60	VCC	P	-	-	-
61	GND	P	-	-	-
62	EMIT_11	O	±X	-	-
63	EMIT_12	O	±X	-	-
64	VCC	P	-	-	-
65	GND	P	-	-	-
66	P3_1/TXD0	I/O	±5	PU	-
67	P3_0/RXD0	I/O	±5	PU	-
68	P4_5/MISO1/INTB3	I/O	±5	PU	-
69	P4_4/MOSI1/INTB2	I/O	±5	PU	-
70	P4_3/SPICLK1/INTB1	I/O	±5	PU	-
71	P4_2/SS1/INTB0	I/O	±5	PU	-
72	P4_1/TXD2	I/O	±5	PU	-
73	P4_0/RXD2	I/O	±5	PU	-
74	VCC	P	-	-	-
75	GND	P	-	-	-
76	INTC3	I	±5	-	-

Table 2-1. **ATPL00B 144-Lead LQFP pinout (Continued)**

PinNo	Pin Name	I/O	I(mA)	Res	HY
77	INTC2	I	±5	-	-
78	INTC1	I	±5	-	-
79	INTC0	I	±5	-	-
80	NC	-	-	-	-
81	NC	-	-	-	-
82	NC	-	-	-	-
83	NC	-	-	-	-
84	NC	-	-	-	-
85	NC	-	-	-	-
86	NC	-	-	-	-
87	NC	-	-	-	-
88	NC	-	-	-	-
89	VDD	P	-	-	-
90	VCC	P	-	-	-
91	GND	P	-	-	-
92	NC	-	-	-	-
93	NC	-	-	-	-
94	NC	-	-	-	-
95	NC	-	-	-	-
96	NC	-	-	-	-
97	NC	-	-	-	-
98	NC	-	-	-	-
99	NC	-	-	-	-
100	NC	-	-	-	-
101	NC	-	-	-	-
102	NC	-	-	-	-
103	VCC	P	-	-	-
104	GND	P	-	-	-
105	INTD3	I	±5	-	-
106	INTD2	I	±5	-	-
107	INTD1	I	±5	-	-
108	INTD0	I	±5	-	-
109	NC	-	-	-	-
110	NC	-	-	-	-
111	NC	-	-	-	-

PinNo	Pin Name	I/O	I(mA)	Res	HY
112	NC	-	-	-	-
113	GND	P	-	-	-
114	DC_COMP	O	±10	-	-
115	VCC	P	-	-	-
116	ENABLE	O	±10	-	-
117	GND	P	-	-	-
118	DNIN	I	±5	-	-
119	DIN	I	±5	-	-
120	REC_1	O	±5	-	-
121	REC_2	O	±5	-	-
122	REC_3	O	±5	-	-
123	REC_4	O	±5	-	-
124	REC_5	O	±5	-	-
125	REC_6	O	±5	-	-
126	REC_7	O	±5	-	-
127	REC_8	O	±5	-	-
128	VCC	P	-	-	-
129	GND	P	-	-	-
130	VRL	I	(**)	-	-
131	VIN	I	(**)	-	-
132	VRH	I	(**)	-	-
133	AVD1	P	-	-	-
134	AVS1	P	-	-	-
135	AVD2	P	-	-	-
136	AVS2	P	-	-	-
137	VCC	P	-	-	-
138	GND	P	-	-	-
139	P5_3/MISO0	I/O	±5	PU	-
140	P5_2/MOSI0	I/O	±5	PU	-
141	P5_1/SPICLK0	I/O	±5	PU	-
142	P5_0/SS0	I/O	±5	PU	-
143	P3_5/T1	I/O	±5	PU	-
144	P3_4/T0	I/O	±5	PU	-

Notes: 1. Mandatory to be tied down

I/O=pin direction: **I**=input, **O**=Output, **P**=Power

I(mA)=nominal current: **+**=source, **-**=sink, **X**=fixed by external resistor

RES=pin pullup/pulldown resistor: **PU**=pullup, **PD**=pulldown ; **HY**=Input Hysteresis

3. Pin Description

Table 3-1. **Pin Description List**

Pin Number	Pin Name	Type	Comments
1	P3.3/INT1	I/O	Microcontroller port 3.3 / External Interrupt 1 • When configured as P3.3, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as INT1, this pin is the 8051C3A microcontroller external interrupt 1 • Internal configuration: 33kΩ typ. pull-up resistor
2, 17, 21, 30, 43, 48, 54, 60, 64, 74, 90, 103, 115, 128, 137	VCC	Power	3.3v digital supply. Digital power supply must be decoupled by external capacitors
3, 4, 5, 19, 22, 23, 29, 44, 49, 55, 61, 65, 75, 91, 104, 113, 117, 129, 138	GND	Power	Digital ground
6	TDI(1)	Input	Test Data In • Internal configuration: 33kΩ typ. pull-up resistor
7	TDO(1)	Output	Test Data out
8	TCK(1)	Input	Test Clock
9	TMS(1)	Input	Test Mode Select • Internal configuration: 33kΩ typ. pull-up resistor
10	TRST(1)	Input	Test Reset • Internal configuration: 33kΩ typ. pull-up resistor
11	D_INIT	Input	Initialization Signal • During power-on, D_INIT should be released before asynchronous reset signal RSTA, in order to ensure proper system start up. Not minimum time is required between both releases, $\Delta t>0$ • D_INIT is active high • Internal configuration: 33kΩ typ. pull-down resistor
12	RSTA	Input	Asynchronous reset • RSTA is a digital input pin used to perform a hardware reset of the ASIC • RSTA is active high • Internal configuration: 33kΩ typ. pull-down resistor

13	/PROG	Input	SPI Flash programming pin
			• /PROG digital input is read during power up and sets the system into “execution” mode or “serial flash programming” mode
			  • ‘0’: Serial flash programming mode
			  • ‘1’: Execution mode (normal mode)
			Internal configuration: 33kΩ typ. pull-up resistor

Table 3-1. **Pin Description List (Continued)**

Pin Number	Pin Name	Type	Comments

14	SECURED	Input	Encryption enable
			• SECURED digital input enables encrypted firmware storage and execution when the board configuration supports it
			  • '0': Encrypted storage/execution disabled
			  • '1': Encrypted storage/execution enabled
			• Internal configuration: 33kΩ typ. pull-down resistor

15	/EWDG	Input	Watchdog enable
			• /EWDG digital input enables watchdog timer. This pin is internally connected to the /EW signal of the ADD8051C3A microcontroller
			  • '0': Watchdog timer enabled
			  • '1': Watchdog timer disabled
			• Internal configuration: 33kΩ typ. pull-down resistor

16	DEBUG	Input	Debug mode enable
			• DEBUG digital input is internally connected to DBG signal of the ADD8051C3A microcontroller, and it is intended to implement software debugging tools
			  • '0': Debug mode disabled
			  • '1': Debug mode enabled
			• Internal configuration: 33kΩ typ. pull-down resistor

18	CLKEB[(2)]	I/O	External clock reference • CLKEB must be connected to one terminal of a crystal (when a crystal is being used) or to one terminal of a compatible oscillator (when a compatible oscillator is being used)
20	CLKEA[(2)]	Input	External clock reference • CLKEA must be connected to one terminal of a crystal (when a crystal is being used) or tied to ground if a compatible oscillator is being used
24, 25	VDEO	Power	LDO 3.3v power supply
26	VSSO	Power	LDO ground
27	LDO_PD	Power	LDO Power-down. This digital input is used to put the internal linear regulator into power down mode • '0': Power down mode disabled '1': Power down mode enabled
28,89	VDD	Power	LDO Power output A capacitor in the range 0.1µF - 10µF must be connected to each pin
31	Vref1	Input	Voltage reference 1. Tie to Vcc with a Rpu = 4.7kΩ
32	Vref2	Input	Voltage reference 2. Tie to GND with a Rpd = 4.7kΩ
33	VNR	Input	Zero-crossing detection signal • This input detects the zero-crossing of the mains voltage, needed to properly determine switching times • Depending on whether an isolated or a non-isolated power supply is being used, isolation of this pin should be taken into account in the circuitry design

Table 3-1. **Pin Description List (Continued)**

Pin Number	Pin Name	Type	Comments
34, 35, 36, 37	TRIAC(3:0)	Output	TRIAC outputs • These four pins are outputs to control home automation devices by the ATPL00B dimmer peripheral. TRIAC outputs can be configured to work as phase angle controllers or as PWM controllers
38	P5.5/TxD1/INTA1	I/O	Microcontroller port 5.5 / Standard Serial Port 1 Tx / Dimmer switch 1 • When configured as P5.5, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as TxD1, this pin is the digital output of the asynchronous standard serial port 1 • When configured as INTA1, this pin is the input to dimmer peripheral signal INTERR(1) • Internal configuration: 33kΩ typ. pull-up resistor
39	P5.4/RxD1/INTA0	I/O	Microcontroller port 5.4 / Standard Serial Port 1 Rx / Dimmer switch 0 • When configured as P5.4, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as RxD1, this pin is the digital input of the asynchronous standard serial port 1 • When configured as INTA0, this pin is the input to dimmer peripheral signal INTERR(0) • Internal configuration: 33kΩ typ. pull-up resistor
40	P4.7/T2EX/INTA3	I/O	Microcontroller port 4.7 / T2EX / Dimmer switch 3 • When configured as P4.7, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as T2EX, this pin is the Timer/Counter 2 capture/reload trigger described in Timer2 section • When configured as INTA3, this pin is the input to dimmer peripheral signal INTERR(3) • Internal configuration: 33kΩ typ. pull-up resistor
41	P4.6/T2/INTA2	I/O	Microcontroller port 4.6 / T2 / Dimmer switch 2 • When configured as P4.6, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as T2, this pin works as the external T2 pin described in Timer2 section • When configured as INTA2, this pin is the input to dimmer peripheral signal INTERR(2) • Internal configuration: 33kΩ typ. pull-up resistor
42	P1.7/SSN	I/O	Microcontroller port 1.7 / Silicon Serial Number • When configured as P1.7, this pin is a pseudo-bidirectional microcontroller I/O port • This pin is the digital input used to read a Serial number if a valid SSN device is being used. This Serial Number is used for encryption purposes. Precaution should be taken if used as generic control port since it searches for a Silicon Serial Number device at start-up and could put out undesirable transient values • Internal configuration: 33kΩ typ. pull-up resistor

Table 3-1. **Pin Description List (Continued)**

Pin Number	Pin Name	Type	Comments
45	EMIT.0	Output	Tx/Rx control output pin • This output pin is used by the system to adapt the external Analog-Front-end either in transmission or in reception, thus improving the electrical behavior
46, 47, 50, 51, 52, 53, 56, 57, 58, 59, 62, 63	EMIT(1:12)	Output	PLC transmission ports[3]
66	P3.1/TxD0	I/O	Microcontroller port 3.1 / Standard Serial Port 0 Tx • When configured as P3.1, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as TxD0, this pin is the digital output of the asynchronous standard serial port 0 • Internal configuration: 33kΩ typ. pull-up resistor
67	P3.0/RxD0	I/O	Microcontroller port 3.0 / Standard Serial Port 0 Rx • When configured as P3.0, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as RxD0, this pin is the digital input of the asynchronous standard serial port 0 • Internal configuration: 33kΩ typ. pull-up resistor
68	P4.5/MISO1/INTB3	I/O	Microcontroller port 4.5 / SPI1 Master In Slave Out / Dimmer switch 3 • When configured as P4.5, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as MISO1, this pin is the SPI1 Master In Slave Out • When configured as INTB3, this pin is the input to dimmer peripheral signal INTERR(3) • Internal configuration: 33kΩ typ. pull-up resistor
69	P4.4/MOSI1/INTB2	I/O	Microcontroller port 4.4 / SPI1 Master Out Slave In / Dimmer switch 2 • When configured as P4.2, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as MOSI1, this pin is the SPI1 Master Out Slave In • When configured as INTB2, this pin is the input to dimmer peripheral signal INTERR(2) • Internal configuration: 33kΩ typ. pull-up resistor
70	P4.3/SPICLK1/INTB1	I/O	Microcontroller port 4.3 / SPI1 Clock / Dimmer switch1 • When configured as P4.3, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as SPICLK1, this pin is the SPI1 clock signal • When configured as INTB1, this pin is the input to dimmer peripheral signal INTERR(1) • Internal configuration: 33kΩ typ. pull-up resistor

Table 3-1. **Pin Description List (Continued)**

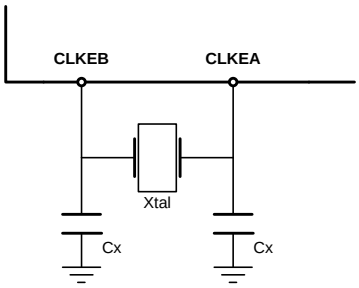
Pin Number	Pin Name	Type	Comments
71	P4.2/SS1/INTB0	I/O	Microcontroller port 4.2 / SPI1 Slave Select / Dimmer switch 0 • When configured as P4.2, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as SS1, this pin is the SPI1 Slave Select. Active low • When configured as INTB0, this pin is the input to dimmer peripheral signal INTERR(0) • Internal configuration: 33kΩ typ. pull-up resistor
73	P4.0/RxD2	I/O	Microcontroller port 4.0 / Standard Serial Port 2 Rx • When configured as P4.0, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as RxD2, this pin is the digital input of the asynchronous standard serial port 2 • Internal configuration: 33kΩ typ. pull-up resistor
76, 77, 78, 79	INTC(3:0)	I/O	Dimmer switches • • These pins can be configured as inputs to dimmer peripheral signals INTERR(3:0)
80, 81, 82, 83, 84, 85, 86, 87, 88, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 109, 110, 111, 112	NC	-	No Connect
105, 106, 107, 108	INTD(3:0)	I/O	Dimmer switches • • These pins can be configured as inputs to dimmer peripheral signals INTERR(3:0)
114	DC_COMP	Output	External comparator DC compensation • This output controls the direct current compensation of the external comparator. This pin is connected only if an external comparator is being used. Otherwise, this pin must be left unconnected
116	ENABLE	Output	External comparator enable • Output pin to enable external comparator. This pin is connected only if an external comparator is being used. Otherwise, this pin must be left unconnected
118	DNIN	Input	External comparator inverted output signal • This pin is the input for the external comparator inverted output signal. This pin is connected only if an external comparator is being used. Otherwise, this pin must be tied to ground
119	DIN	Input	External comparator output signal • This pin is the input for the external comparator output signal. This pin is connected only if an external comparator is being used. Otherwise, this pin must be tied to ground

Table 3-1. **Pin Description List (Continued)**

Pin Number	Pin Name	Type	Comments
120, 121, 122, 123, 124, 125, 126, 127	REC(1:8)	Output	External comparator threshold loop • These outputs are used to set the voltage threshold in the external comparator. These pins are connected only if an external comparator is being used. Otherwise, these pins must be left unconnected
130	VRL	Input	Analog input low voltage reference
131	VIN	Input	Direct-analog input voltage
132	VRH	Input	Analog input high voltage reference
133, 135	AVD1, AVD2	Power	3.3v analog power
134, 136	AVS1, AVS2	Power	Analog ground
139	P5.3/MISO0	I/O	Microcontroller port 5.3 / SPI0 Master In Slave Out • When configured as P5.3, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as MISO0, this pin is the SPI0 Master In Slave Out • Internal configuration: 33kΩ typ. pull-up resistor
140	P5.2/MOSI0	I/O	Microcontroller port 5.2 / SPI0 Master Out Slave In • When configured as P5.2, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as MOSI0, this pin is the SPI0 Master Out Slave In • Internal configuration: 33kΩ typ. pull-up resistor
141	P5.1/SPICLK0	I/O	Microcontroller port 5.1 / SPI0 Clock • When configured as 54.1, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as SPICLK0, this pin is the SPI0 clock signal • Internal configuration: 33kΩ typ. pull-up resistor
142	P5.0/SS0	I/O	Microcontroller port 5.0 / SPI0 Slave Select • When configured as P5.0, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as SS0, this pin is the SPI0 Slave Select. Active low • Internal configuration: 33kΩ typ. pull-up resistor
143	P3.5/T1	I/O	Microcontroller port 3.5 / Timer 1 interrupt signal • When configured as P3.5, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as T1, this pin works as the external T1 pin described in Timer1 section • Internal configuration: 33kΩ typ. pull-up resistor
144	P3.4/T0	I/O	Microcontroller port 3.4 / Timer 0 interrupt signal • When configured as P3.4, this pin is a pseudo-bidirectional microcontroller I/O port • When configured as T0, this pin works as the external T0 pin described in Timer0 section • Internal configuration: 33kΩ typ. pull-up resistor

Notes: 1. This pin is part of the JTAG Boundary Scan interface and is only used for boundary scan purposes

2. The crystal should be located as close as possible to CLKEA and CLKEB pins. Recommended value for Cx is 18pF. This value may depend on the specific crystal characteristics



Different configurations allowed depending on external topology and net behavior

4. Processor and Architecture

4.1 ADD8051C3A Microcontroller Description

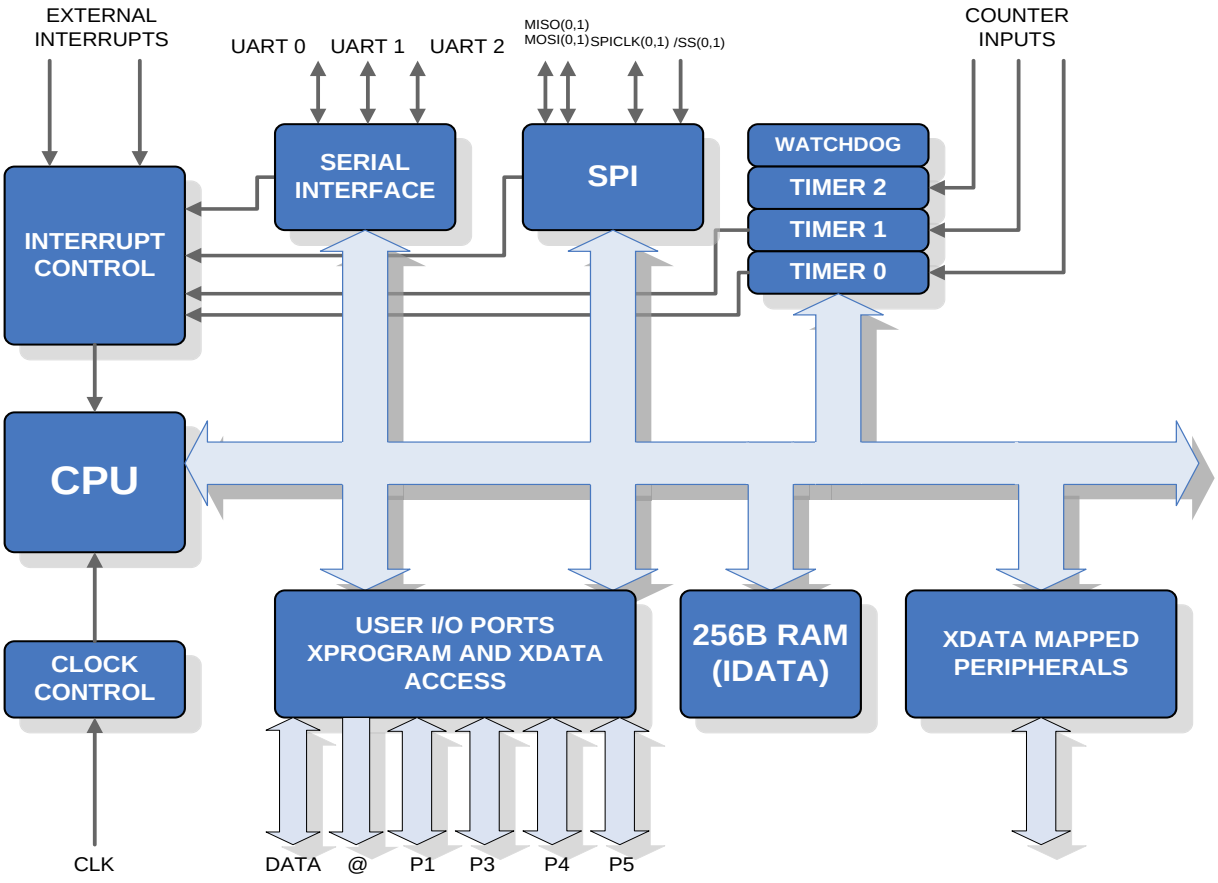
The following document provides a detailed description of the ADD8051C3A architecture and its hardware. ADD8051C3A is fully compatible with any 8051 legacy microcontroller, however there are some hardware differences that deserve to be taken into account.

CPU speed has been improved. The machine cycle has been reduced from 12 to 3 clock cycles, and all the instructions are executed in 1 or 2 machine cycles. Programs are executed a minimum of four times faster than in a standard-8051-architecture device, achieving x5 speedup for most programs.

A 128KBytes SRAM is embedded on chip. Program memory and External Data (XDATA) memory share this 128Kbytes SRAM. The lower 64Kbytes are always dedicated to store program memory, while upper 64Kbytes are configurable to allocate different program and XDATA memory size configurations, depending on bank switching page size (See 4.3).

This document includes a detailed description of registers and peripherals. Some of these items are upgraded versions of the original 8051 microcontroller circuitry, while others are a simplified version.

Figure 4-1. ADD8051C3A microcontroller block diagram



Note: This chapter describes the architecture from the microcontroller point of view. Thus, “on-chip SRAM” is referred to as “external memory” since “external” instructions like MOVX are necessary to access this memory device

4.2 Core Pinout Description

Figure 4-2. ADD8051C3A core pinout diagram

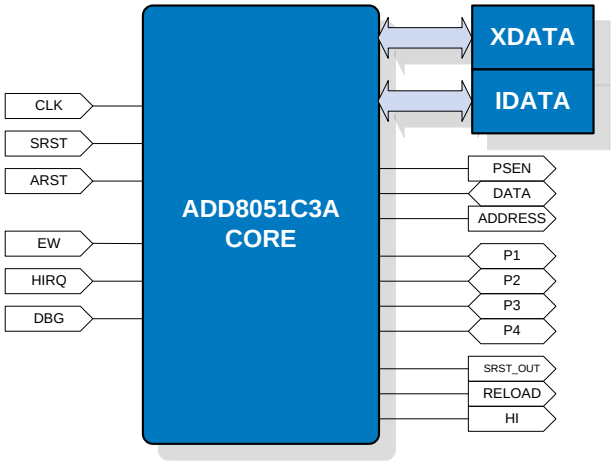


Table 4-1. **Core pinout description**

Pin	I/O	Description
CLK	I	Clock input.
ARST	I	Asynchronous reset.
SRST	I	SRST Must be held active one clock cycle to ensure proper power-on. External signal D_INIT is connected to this core input.
/EW	I	Watchdog enable Low level active. External signal/EWDG is connected to this core input.
HIRQ	I	Hard idle request High level active. When asserted, the microcontroller finishes the instruction in execution and goes to a special hard idle state. In hard idle state the CPU is stopped while peripheral circuits still run, CPU restarts only after HIRQ deassertion.
DBG	I	Debug mode enable High level active.
PSEN	O	Program Store Enable Read strobe to program memory.
DATA	I/O	Data bus 16-bit bidirectional port to access program and/or data memories.
ADDRESS	O	Address bus Output port to access program memory and XDATA.
SRST_OUT	O	Watchdog timer reset signal.
RELOAD	O	Program reload control signal Software or watchdog activated. Force program reload from a serial storage device to a parallel SRAM.
HI	O	Hard idle flag, This flag is set to ‘1’ when the CPU is in hard idle state.

PORTS: P1, P3, P4, P5 (*)	I/O	I/O Ports. 8-bit pseudo bidirectional I/O ports with pull-up resistors. These ports are by default in **pseudo-bidirectional** configuration. Configured in this way, port pins that have 1s written to them are pulled high by the internal pull-ups, and in that state can be used as inputs. When a bit in a Port register has a 0 to 1 transition the related pin is driven high using transistor during 1 clock cycle, then the transistor is switched off and the pull-up resistor keeps the logic level. Ports P3, P4 and P5 can be also configured as **push-pull** mode ports. In push-pull mode the pin is always in output mode and logic 1 is always high driven (see 4.7.1). Some pins of P3, P4 and P5 also serve the functions of various special features of the microcontroller: P3.0 → RxD (serial port 0 input) P3.1 → TxD (serial port 0 output) P3.2 → INT0 (external interrupt 0) P3.3 → INT1 (external interrupt 1) P3.4 → T0 (timer 0 external input) P3.5 → T1 (timer 1 external input) P3.6 → WR(external data memory write strobe) P3.7 → RD (external data memory read strobe) P4.0 → RxD (serial port 2 input) P4.1 → TxD (serial port 2 output) P4.2 → SS1 (slave select input) P4.3 → SPICLK1 (SPI1 clock input/output) P4.4 → MOSI1 (master out / slave in data) P4.5 → MISO1 (master in / slave out data P4.6 → T2 (timer 2 input/output) P4.7 → T2EX (timer 2 external input) P5.0 → SS0 (slave select input) P5.1 → SPICLK0 (SPI0 clock input/output) P5.2 → MOSI0 (master out / slave in data) P5.3 → MISO0 (master in / slave out data) P5.4 → RxD (serial port 1 input) P5.5 → TxD (serial port 1 output) (*) each bit of ports 1, 3, 4 and 5 is composed of three lines: data output, data input and output enable (low level active)

4.3 Memory Organization

ADD8051C3A has separate logical address spaces for program and data memory, as shown in Figure 4-3. The logical separation of program and data memory allows the data memory to be accessed by 8-bit addresses, which can be quickly stored and manipulated by an 8-bit CPU. Nevertheless, 16-bit data memory addresses can also be generated through the DPTR (Data Pointer Registers).

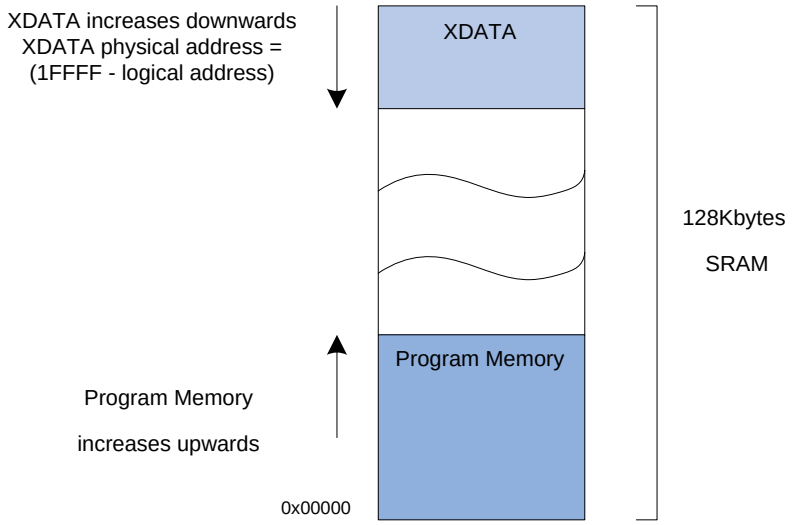
Program Memory and XDATA Memory are combined in a single 128Kbytes SRAM and accessed by the same bus using internal RD and PSEN signals functions.

- Program memory must only be read. Writing in Program Memory could lead to unexpected results and must be avoided. There is a block of 64kbytes of base program memory, which can be increased by extended program pages.

- XDATA occupies a separate address space from Program Memory. Up to 64kbytes of XDATA can be addressed in the Data Memory space. The CPU generates the internal “read” and “write” signals, RD and WR, needed during XDATA Memory accesses.

Additionally to these 128 Kbytes of SRAM, there are 384 bytes of internal data memory (IDATA) in the ADD8051C3A core. This is explained in more detail in 4.3.3 section.

Figure 4-3. 128KB SRAM, XDATA and Program Memory



The program must be uploaded to the SRAM after power up. The source of program must be a non volatile storage unit (i.e. serial flash). A specific module (boot loader peripheral) has been designed to control the booting of the system after power up. The boot loader module can also be used to program the non volatile storage unit (serial flash) using a serial link. The non volatile storage unit must be connected to P5 port pins used for the SPI0 link (see Table 4-1). The serial link to the boot loader shares pins with the microcontroller Serial Port 0. For more information see boot loader section in 8

4.3.1 Program Memory

After reset, the CPU begins execution from location 0000H and stack pointer (SP) value is set to 07H.

Interrupt Handling

A fixed location in Program Memory is assigned to each interrupt. The interrupt causes the CPU to jump to that location, where it begins executing the service routine.

Example: External Interrupt 0, is assigned to location 0003H. If External Interrupt 0 is going to be used, its service routine must begin at location 0003H. If the interrupt is not going to be used, its service location is available as general purpose Program Memory.

The interrupt service locations are spaced at 8-byte intervals beginning at 0x0003 (i.e.0x0003 for External Interrupt0, 0x000B for Timer0, etc.).

If an interrupt service routine is short enough, it can reside entirely within that 8-byte interval. Longer service routines can use a jump instruction to skip over subsequent interrupt locations, if other interrupts are in use.

Program Memory addresses are always 17 bits wide, even though the actual amount of Program Memory used may be less than 64kbytes.

P0 and P2 ports are no longer used to access external memories, but Special Function Registers (SFR) P0 and P2 are still functional.

- P2 register is used to generate the high order address byte when executing MOVX A,@Ri or MOVX @Ri,A.
- P0 register is the program address control register and it is used to specify the size of extended program pages and to control bank switching.

4.3.2 Extended Addressing

ATPL00B combines program code and data in a single 128KB SRAM.

ADDRESS to access program is generated as a function of P0(7:6), P0(5:0) and PC(15:0).

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
P0	SX1	SX0	P05	P04	P03	P02	P01	P00

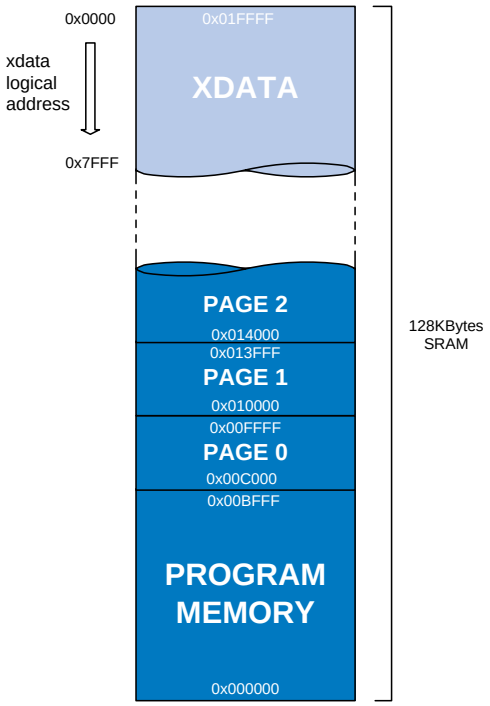
- SX(1:0): Size of extended program pages and common program space

SX(1:0)	Extended	Common
“11”	0KB	64KB
“10”	8KB	56KB
“01”	16KB	48KB
“00”	32KB	32KB

- P0(5:0): Extended program page number

Allowing different configurations as the one shown in Figure 4-4

Figure 4-4. SRAM Extended Addressing. 48KB common memory + 3x16KB extended program pages



Common and extended memory sizes vary depending on the value in P0(7:6), as shown above. According to these size values, the core knows when the PC is pointing to an address located in common memory and when it is pointing to an address located in extended memory. When PC is pointing to an address in extended memory, then P0(5:0) indicates the extended page number and the physical address is automatically calculated. Some examples are shown below.

Example: SX(1:0)=”00” (common memory size = 32KB)

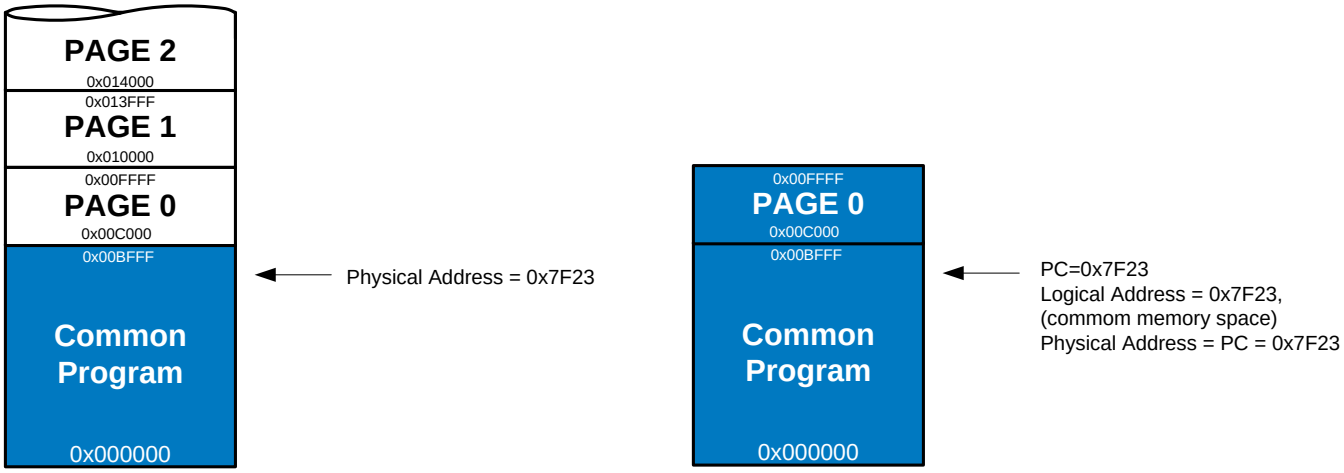
PC=0x7F23 (PC pointing to an address in common memory space)

P0(5:0)= don’t care

LOGIC ADDRESS=0x7F23, common memory space

PHYSICAL ADDRESS=0x7F23 (common memory space → P0(5:0) is ignored)

Figure 4-5. Extended Addressing example 1



Example: SX(1:0)=”00” (common memory size = 32KB)

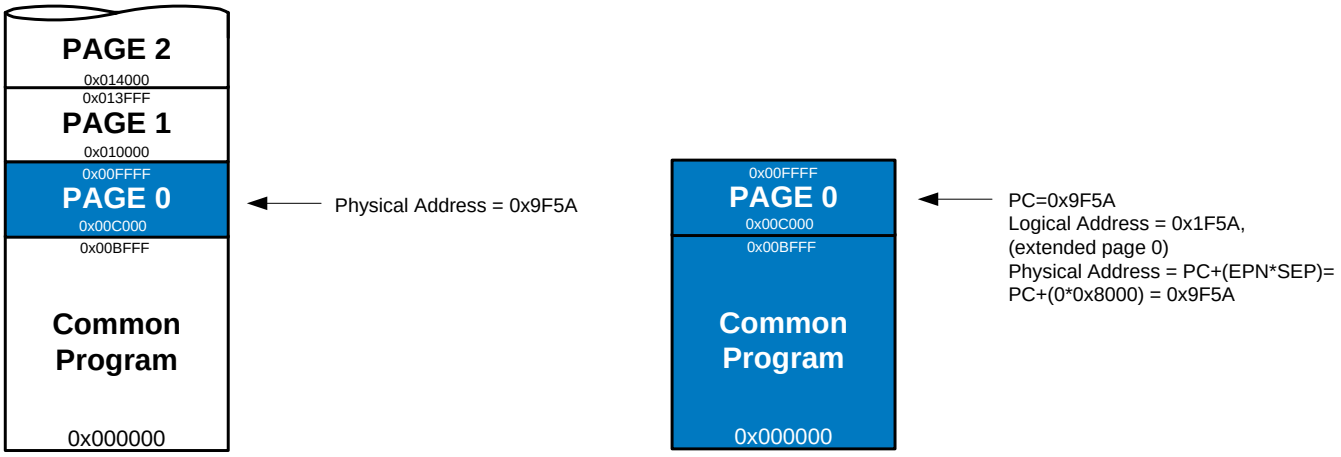
PC=0x9F5A (PC pointing to an address in extended memory)

P0(5:0)= 0x00 (extended page 0)

LOGIC ADDRESS=0x1F5A, extended page 0

PHYSICAL ADDRESS=0x9F5A

Figure 4-6. Extended Addressing example 2



Example: SX(1:0)=”00” (common memory size = 32KB)

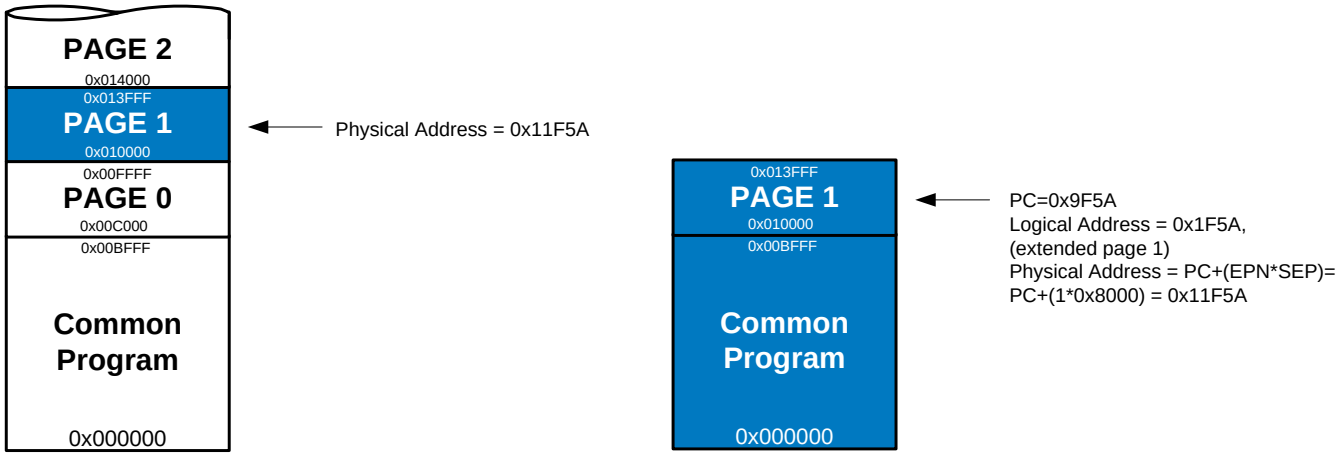
PC=0x9F5A (PC pointing to an address in extended memory)

P0(5:0)= 0x01 (extended page 1)

LOGIC ADDRESS=0x1F5A, extended page 1

PHYSICAL ADDRESS=0x11F5A

Figure 4-7. Extended Addressing example 3



Thus if PC < SCS then Physical Address = PC

else Physical Address = PC+(EPN*SEP)

where SCS= Size of Common Space;

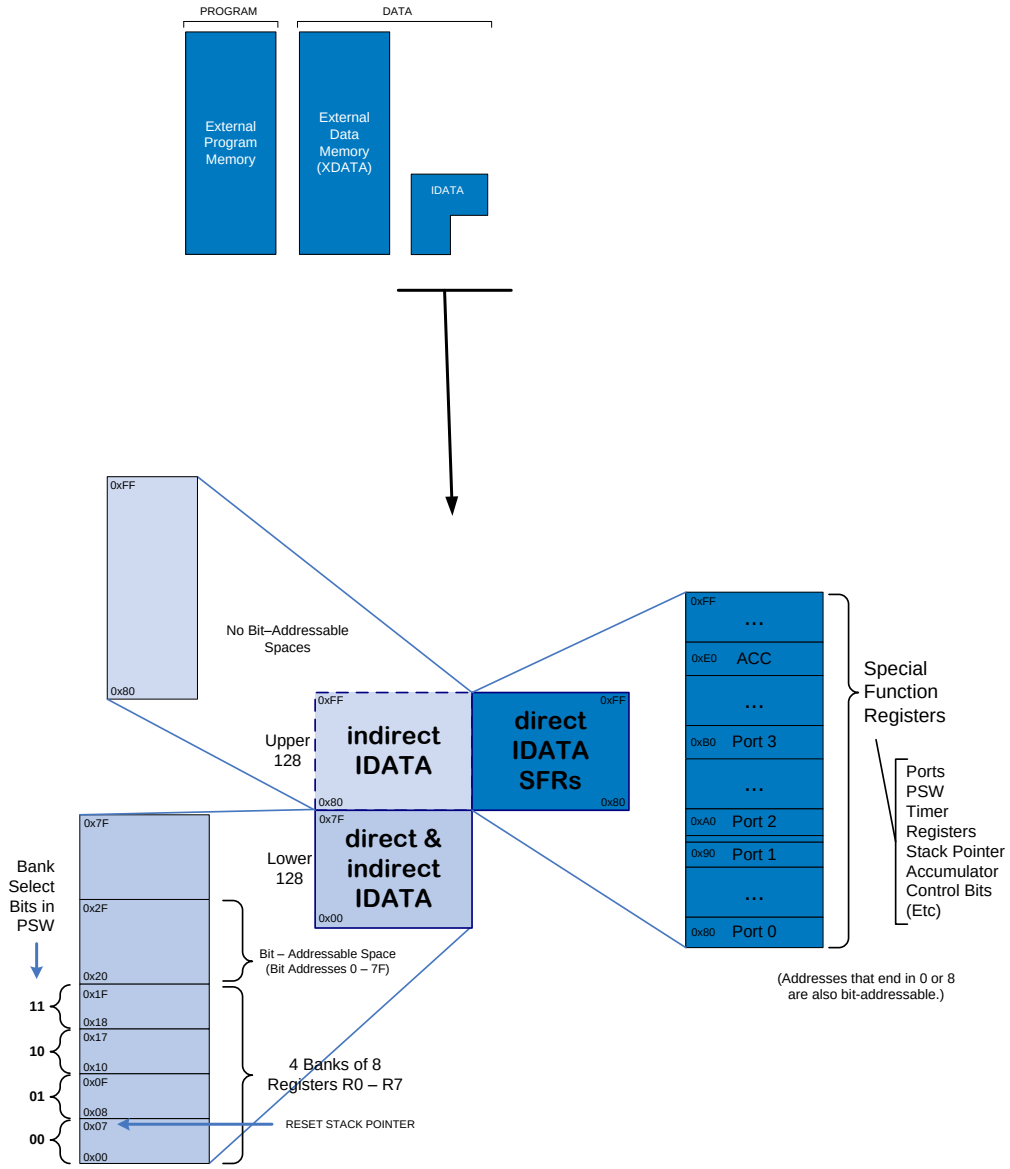
EPN=Extended Page Number;

SEP= Size of Extended Pages(defined by SX(1:0) value).

4.3.3 Data Memory

Figure 4-8 shows the internal and external Logical Data Memory spaces available to the microcontroller user.

Figure 4-8. Data Memory Space



Logical addressing of the Internal Data Memory is mapped as shown in Figure 4-8**.** The internal IDATA memory space has a total size of 384 bytes and is divided into three blocks, which are generally referred to as the Lower 128 bytes, the Upper 128 bytes and SFR space (128 bytes size).

Internal Data Memory addresses are always one byte wide, which implies an address space of only 256 bytes. A simple trick is used to accommodate the 384 bytes of IDATA using 8 bit addresses: direct addresses higher than 7FH access SFR memory space, whereas indirect addresses higher than 7FH access the Upper 128 bytes of IDATA. Thus, Figure 4-8 shows the Upper 128 bytes and SFR space occupying the same block of addresses, 80H through FFH, although they are physically separate entities.

In the Lower 128 bytes of IDATA, the lowest 32 bytes are grouped into 4 banks of 8 registers. Program instructions call out these registers as R0 through R7. Two bits in the Program Status Word (PSW) select which register bank is in use. This allows more efficient use of code space, since register instructions are shorter than instructions that use direct addressing.

The next 16 bytes above the register banks form a block of bit-addressable memory space. The 80C51 instruction set includes a wide selection of single-bit instructions, and the 128 bits in this area can be directly addressed by these instructions. The bit addresses in this area are 00H through 7FH.

Summarizing, the Lower 128 bytes are accessed independently of the addressing mode (direct or indirect addressing). The Upper 128 bytes can be accessed only by indirect addressing, whereas SRF space is accessed only by direct addressing.

Table 4-2 gives a look at the Special Function Register (SFR) space. SFRs include the Port registers, timers, peripheral controls, etc. These registers can only be accessed by direct addressing. Sixteen addresses in SFR space are both byte and bit addressable. The bit addressable SFRs are those whose address ends in 0H or 8H. ADD8051C3A includes the same 21 SFRs included in the legacy 8051C plus additional SFRs to implement the extended features.

The CPU generates the RD (P3.7) and WR (P3.6) signals needed during SRAM accesses, as shown below.

4.3.4 SFR Registers

Table 4-2 shows a map of the internal memory area called the Special Function Register (SFR) space. The entries in bold are registers not included in the standard architecture of the 8051, these entries are specific for the ADD8051C3A. Blank entries are not implemented on the chip. User software should not write to these unimplemented locations. Read accesses to these addresses will return random data

Table 4-2. **Special Function Registers (specific ADD8051C3A registers in bold)**

F8								**T3**	FF
F0	B	**CONF**							F7
E8									EF
E0	ACC	**SPSTAT**	**SPCTL**	**SPDAT**	**SPSTAT1**	**SPCTL1**			E7
D8	**SPDAT10**	**SPDAT11**	**SPDAT12**	**SPDAT13**	**SPDAT14**	**SPDAT15**	**SPDAT16**	**SPDAT17**	DF
D0	PSW								D7
C8	**T2CON**	**T2MOD**	**RCAP2L**	**RCAP2H**	**TL2**	**TH2**			CF
C0	**P4**				**P5**				C7
B8	IP								BF
B0	P3							**IPH**	B7
A8	IE								AF
A0	P2	**IE2**	**AUX1**	**AUX2**					A7
98	SCON	SBUF	**SCON1**	**SBUF1**	**P3M**	**P4M**	**P5M**		9F
90	P1	**IP2HL**			**SCON2**	**SBUF2**	**T1NP**		97
88	TCON	TMOD	TL0	TL1	TH0	TH1	**TPR**	**T1NC**	8F
80	P0	SP	DPL	DPH				PCON	87

The functions of some SFRs are described in the text below, while others are described with their related peripherals.

- **Accumulator (address: E0H**)

  ACC is the Accumulator register. Note that mnemonics for accumulator specific instructions usually refer to the Accumulator simply as A.

- **B Register (address: F0H)**

  The B register is used during multiply and divide operations. For other instructions it can be treated as another scratch pad register.

- **PSW register (address: D0H)**

  The PSW register contains program status information as detailed below.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
PSW	CY	AC	F0	RS1	RS0	OV	F1	P

  - CY: Carry flag
  - AC: Auxiliary carry flag
  - F0: General purpose flag
  - RS(1:0): Register bank select control bits
  - OV: Overflow flag
  - F1: User definable flag
  - P: Parity flag

- **Stack Pointer (address: 81H)**

  8 bit stack pointer that is initialized to internal memory of 07H. The user program can initialize it at any internal RAM location ranging from 07H to FFH.

- **Auxiliary 1**

  The AUX1 register includes the data pointer selection bit, the software reset control and the switch to enable wake-up from power-down using external interrupts.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
AUX1	--	--	SRST	GF2	WUPD	0	--	DPS

  - --: Reserved bits
  - SRST: Software reset
  - GF2: General purpose flag
  - WUPD: When set, enables external interrupts driven wake-up from power down
  - 0: fixed ‘0’
  - DPS: Data pointer selector, selects between DPTR0 and DPTR1

- **P0, P1, P2, P3, P4, P5 registers**

  P1, P3, P4 and P5 are the SFR registers of Ports 1, 3, 4 and 5 respectively.

  Writing a one/zero to a bit in these registers causes the corresponding port output pin to switch high/low. When a port bit is used as an input the corresponding port SFR must be set to ‘1’.

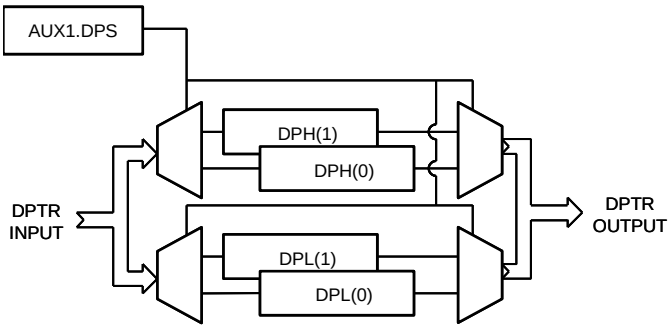
  Ports 3, 4 and 5 are pseudo bidirectional by default, and can be configured to push-pull or pseudo bidirectional using SFRs P3M, P4M and P5M respectively, as follows: when P4M(i) is set, P4(i) is configured as push-pull.

P0 and P2 ports are no longer used to address program and external data. P0 SFR takes control of extended addressing and program banking and P2 SFR is used when MOVX @Ri instructions are executed (in this case, P2 SFR content is used as most significant address byte).

- **Data Pointer**

  The Data Pointer Register (DPTR) consists of a high byte (DPH) and a low byte (DPL). Its intended function is to hold a 16-bit address. It may be manipulated as a 16-bit register or as two independent 8-bit registers. The same entry accesses two different data pointers (DPTR0, DPTR1), user software can switch between them using the data pointer selection bit in AUX1.

Figure 4-9. DPTR selection scheme



- **Serial Data Buffers**

  The Serial Buffers have actually two separate registers, a transmit buffer and a receive buffer. When data is moved to SBUF, it goes to the transmit buffer and is held for serial transmission. (Moving a byte to SBUF is what initiates the transmission.) When data is moved from SBUF, it comes from the receive buffer.

- **Timer Registers**

  Register pairs (TH0, TL0), (TH1, TL1) and (TH2, TL2) are the 16-bit Counting registers for Timer/Counters 0, 1 and 2, respectively.

  T3 is the 8-bit counting register of the watchdog timer, see 5.3 for detailed behavior description.

- **CONF Register**

  CONF register includes specific configuration features for the flash loader and Standard Serial Interface.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
CONF	--	--	--	--	--	RLD	RLT3	BAUD2

  - --: Reserved bits
  - RLD: When set, it forces program reload from SPI flash
  - RLT3: When set, it forces program reload from SPI flash when watchdog timeout occurs
  - BAUD2: doubles USART baud rates when set

- **Control Registers**

  Special Function Registers IP, IPH, IP2HL, IE, IE2, TMOD, TCON, T2CON, SCONj, SPSTATj and PCON contain control and status bits for the interrupt system, the Timer/Counters, the serial ports and the serial peripheral interfaces

4.4 Instruction Set

The instruction set provides a variety of fast addressing modes for accessing the internal RAM to facilitate byte operations on small data structures. The instruction set provides extensive support for one-bit variables as a separate data type, allowing direct bit manipulation in control and logic systems that require boolean processing.

4.4.1 Program Status Word

The Program Status Word (PSW) contains several status bits that reflect the current state of the CPU. The PSW, shown in 4.3.4, resides in the SFR space. It contains the Carry bit, the Auxiliary Carry (for BCD operations), the two register bank select bits, the Overflow flag, a Parity bit, and two user-definable status flags.

The Carry bit, other than serving the function of a Carry bit in arithmetic operations, also serves as the “Accumulator” for a number in Boolean operations.

The bits RS0 and RS1 are used to select one of the four register banks shown in Figure 4-8. A number of instructions refer to these RAM locations as R0 through R7. The selection of which of the four is being referred to is made on the basis of the RS0 and RS1 at execution time.

The Parity bit reflects the number of 1s in the Accumulator: P = 1 if the Accumulator contains an odd number of 1s, and P = 0 if the Accumulator contains an even number of 1s. Thus the number of 1s in the Accumulator plus P is always even. Two bits in the PSW are uncommitted and may be used as general purpose status flags.

4.4.2 Addressing Modes

- Direct Addressing

  In direct addressing the operand is specified by an 8-bit address field in the instruction. Only internal Data RAM and SFRs can be directly addressed.

- Indirect Addressing

  In indirect addressing the instruction specifies a register which contains the address of the operand. Both internal and external RAM can be indirectly addressed.

  The address register for 8-bit addresses can be either R0 / R1 of the selected register bank or the Stack Pointer. The address register for 16-bit addresses can only be the 16-bit “data pointer” register, DPTR.

- Register Instructions

  The register banks, containing registers R0 through R7, can be accessed by certain instructions which carry a 3-bit register specification within the opcode of the instruction. Instructions that access the registers this way are code efficient, since this mode eliminates an address byte. When the instruction is executed, one of the eight registers in the selected bank is accessed. One of four banks is selected at execution time by the two bank select bits in the PSW.

- Register-Specific Instructions

  Some instructions are specific to a certain register. For example, some instructions always operate on the Accumulator, or Data Pointer, etc., so no address byte is needed to point to it. The opcode itself does that. Instructions that refer to the Accumulator as A assemble as accumulator specific opcodes.

- Immediate Constants

  The value of a constant can follow the opcode in Program Memory.

  For example:

  MOV A, 64H

  loads the Accumulator with the number 64H.

- Indexed Addressing

  Only program Memory can be accessed with indexed addressing, and it can only be read. This addressing mode is intended for reading look-up tables in Program Memory. A 16-bit base register (either DPTR or the Program Counter) points to the base of the table, and the Accumulator is set up with the table entry number.

  The address of the table entry in Program Memory is formed by adding the Accumulator data to the base pointer.

  Another type of indexed addressing is used in the “case jump” instruction. In this case the destination address of a jump instruction is computed as the sum of the base pointer and the Accumulator data.

4.4.3 Arithmetic Instructions

The menu of arithmetic instructions is listed in Table 4-3, which indicates the addressing modes that can be used with each instruction to access the <byte> operand. For example, the ADD A,<byte> instruction can be written as:

```
ADD    A, 6EH     (direct addressing)
ADD    A, @R1     (indirect addressing)
ADD    A, R6      (register addressing)
ADD    A, #206    (immediate constant)
```

The execution times listed in Table 4-3 assume 3 clock cycles per machine cycle (mc), using a 10MHz clock 1 mc is executed in 0.3us. All the arithmetic instructions execute in 1mc except the Multiply and Divide instructions, which take 2mc.

Note that any byte in the internal Data Memory space can be incremented without going through the Accumulator.

One of the INC instructions operates on the 16-bit Data Pointer. The Data Pointer is used to generate 16-bit addresses for external memory, so being able to increment it in one 16-bit operation is a useful feature.

The MUL AB instruction multiplies the Accumulator by the data in the B register and puts the 16-bit product into the concatenated B and Accumulator registers.

The DIV AB instruction divides the Accumulator by the data in the B register and leaves the 8-bit quotient in the Accumulator, and the 8-bit remainder in the B register. Oddly enough, DIV AB finds less use in arithmetic “divide” routines than in radix conversions and programmable shift operations. In shift operations, dividing a number by 2^n shifts its n bits to the right. Using DIV AB to perform the division completes the shift in 2mc and leaves the B register holding the bits that were shifted out.

The DA A instruction is for BCD arithmetic operations. In BCD arithmetic, ADD and ADDC instructions should always be followed by a DA A operation, to ensure that the result is also in BCD. Note that DA A will not convert a binary number to BCD. The DA A operation produces a meaningful result only as the second step in the addition of two BCD bytes.

Table 4-3. **Arithmetic Instructions**

MNEMONIC	OPERATION	ADDRESSING MODES				EXECUTION
		DIR	IND	REG	IMM	TIMES (mc)
ADD A,<byte>	A = A + <byte>	X	X	X	X	1
ADD A,<byte>	A = A + <byte> + C	X	X	X	X	1
SUBB A,<byte>	A = A – <byte> – C	X	X	X	X	1
INC A	A = A + 1	Accumulator only				1
INC<byte>	<byte> = <byte> + 1	X	X	X		1
INC DPTR	DPTR = DPTR + 1	Data Pointer only				1
DEC A	A = A – 1	Accumulator only				1
DEC<byte>	<byte> = <byte> – 1	X	X	X		1
MUL AB	B*A = B x A	ACC and B only				2
DIV AB	A = Int[A/B]	ACC and B only				2
	B = Mod[A/B]					
DA A	Decimal Adjust	Accumulator only				1

4.4.4 Logical Instructions

Table 4-4 shows the list of logical instructions. The instructions that perform Boolean operations (AND, OR, Exclusive OR, CPL) on bytes perform the operation on a bit-by-bit basis. That is, if the Accumulator contains 00110101B and byte contains 01010011B, then:

ANL A, <byte>

will leave the Accumulator holding 00010001B.

The addressing modes that can be used to access the <byte> operand are listed in Table 4-4.

The ANL A, <byte> instruction may take any of the forms:

```
ANL    A,6FH      (direct addressing)
ANL    A,@R0      (indirect addressing)
ANL    A,R5       (register addressing)
ANL    A,#42H     (immediate constant)
```

All of the logical instructions execute in 1mc (0.3us using a 10MHz clock).

Table 4-4. **Logical Instructions**

MNEMONIC	OPERATION	ADDRESSING MODES				EXECUTION
		DIR	IND	REG	IMM	TIME (mc)
ANL A,<byte>	A = A.AND. <byte>	X	X	X	X	1
ANL <byte>,A	<byte> = <byte> .AND.A	X				1
ANL <byte>,#data	<byte> = <byte> .AND.#data	X				1
ORL A,<byte>	A = A.OR.<byte>	X	X	X	X	1
ORL <byte>,A	<byte> = <byte> .OR.A	X				1
ORL <byte>,#data	<byte> = <byte> .OR.#data	X				1
XRL A,<byte>	A = A.XOR. <byte>	X	X	X	X	1
XRL <byte>,A	<byte> = <byte> .XOR.A	X				1
XRL <byte>,#data	<byte> = <byte> .XOR.#data	X				1
CLR A	A = 00H	Accumulator only				1
CPL A	A = .NOT.A	Accumulator only				1
RL A	Rotate ACC Left 1 bit	Accumulator only				1
RLC A	Rotate Left through Carry	Accumulator only				1
RR A	Rotate ACC Right 1 bit	Accumulator only				1
RRC A	Rotate Right through Carry	Accumulator only				1
SWAP A	Swap Nibbles in A	Accumulator only				1

Note that Boolean operations can be performed on any byte in the internal Data Memory space without going through the Accumulator. The *XRL <byte>, #data instruction*, for example, offers a quick and easy way to invert port bits, as in *XRL P1, #0FFH*.

If the operation is in response to an interrupt, not using the Accumulator saves time and effort to push it onto the stack in the service routine.

The Rotate instructions (RL, A, RLC A, etc.) shift the Accumulator 1 bit to the left or right. For a left rotation, the MSB rolls into the LSB position. For a right rotation, the LSB rolls into the MSB position.

The SWAP A instruction interchanges the high and low nibbles within the Accumulator. This is a useful operation in BCD manipulations.

4.4.5 Data Transfer Instructions

Internal RAM

Table 4-5 shows the menu of instructions that are available for moving data around within the internal memory spaces, and the addressing modes that can be used with each one. All of these instructions are executed in 1 mc.

Table 4-5. **Data Transfer Instructions that Access Internal Data Memory Space**

MNEMONIC	OPERATION	ADDRESSING MODES				EXECUTION TIME (mc)
		DIR	IND	REG	IMM	
MOV A,<src>	A = <src>	X	X	X	X	1
MOV <dest>,A	<dest> = A	X	X	X		1
MOV <dest>,<src>	<dest> = <src>	X	X	X	X	1
MOV DPTR,#data16	DPTR = 16-bit immediate constant				X	1
PUSH <src>	INC SP:MOV“@SP”,<src>	X				1
POP <dest>	MOV <dest>,“@SP”:DEC SP	X				1
XCH A,<byte>	ACC and <byte> exchange data	X	X	X		1
XCHD A,@Ri	ACC and @Ri exchange low nibbles		X			1

The MOV <dest>, <src> instruction allows data to be transferred between any two internal RAM or SFR locations without going through the Accumulator. Remember, the upper 128 bytes of IDATA can be accessed only by indirect addressing, and SFR space only by direct addressing.

Note that in any 80C51 device, the stack resides in on-chip RAM, and grows upwards. The PUSH instruction first increments the Stack Pointer (SP), then copies the byte into the stack. PUSH and POP use only direct addressing to identify the byte being saved or restored, but the stack itself is accessed by indirect addressing using the SP register. This means the stack can go into the Upper 128 bytes of RAM, if they are implemented, but not into SFR space.

The Data Transfer instructions include a 16-bit MOV that can be used to initialize the Data Pointer (DPTR) for look-up tables in Program Memory, or for 16-bit external Data Memory accesses.

The XCH A, <byte> instruction causes the Accumulator and addressed byte to exchange data. The XCHD A, @Ri instruction is similar, but only the low nibbles are involved in the exchange.

External RAM (from microcontroller point of view)

Table 4-6 shows a list of the Data Transfer instructions that access external Data Memory. Only indirect addressing can be used. The choice is whether to use a one-byte address, @Ri, where Ri can be either R0 or R1 of the selected register bank, or a two-byte address, @DPTR. All of these instructions execute in 2 mc.

Note that in all external Data RAM accesses, the Accumulator is always either the destination or source of the data.

The read and write strobes to external RAM are activated only during the execution of a MOVX instruction.

Table 4-6. **Data Transfer Instructions that access External Data Memory Space**

ADDRESS WIDTH	MNEMONIC	OPERATION	EXECUTION TIME (mc)
8 bits	**MOVX A,@Ri**	Read external RAM @Ri	2
8 bits	**MOVX @Ri,A**	Write external RAM @ Ri	2
16 bits	**MOVX A,@DPTR**	Read external RAM @ DPTR	2
16 bits	**MOVX @DPTR,A**	Write external RAM @ DPTR	2

Code Constants

Table 4-7 shows the two instructions that are available for reading code constants in Program Memory. Since these instructions access only Program Memory, the code constants can only be read, not updated.

The mnemonic is MOVC for “move constant”. The first MOVC instruction in Table 4-7 can accommodate a table of up to 256 entries numbered 0 through 255. The number of the desired entry is loaded into the Accumulator, and the Data Pointer is set up to point to the beginning of the table. Then, *MOVC A,@A+DPTR* copies the desired table entry into the Accumulator.

The other MOVC instruction works the same way, excepting that the Program Counter (PC) is used as the base address: *MOVC A,@A+PC*

Table 4-7. **Code Constants Read Instructions**

MNEMONIC	OPERATION	EXECUTION TIME (mc)
MOVC A,@A+DPTR	Read program memory at (A + DPTR)	2
MOVC A,@A+PC	Read program memory at (A + PC)	2

4.4.6 Boolean Instructions

80C51 devices contain a complete Boolean (single-bit) processor. The internal RAM contains 128 addressable bits, and the SFR space can support up to 128 addressable bits as well. All of the port lines are bit-addressable, and each one can be treated as a separate single-bit port. The instructions that access these bits are not just conditional branches, but a complete menu of move, set, clear, complement, OR, and AND instructions. These kinds of bit operations are not easily obtained in other architectures with any amount of byte-oriented software. The instruction set for the Boolean processor is shown in Table 4-8. All bit accesses are by direct addressing.

Bit addresses 00H through 7FH are in the Lower 128, and bit addresses 80H through FFH are in SFR space.

The Carry bit in the PSW is used as the single-bit Accumulator of the Boolean processor. Bit instructions that refer to the Carry bit as C assemble as Carry-specific instructions (CLR C, etc.). The Carry bit also has a direct address, since it resides in the PSW register, which is bit-addressable.

Note that the Boolean instruction set includes ANL and ORL operations, but not the XRL (Exclusive OR) operation.

There is a series of bit-test instructions which execute a jump if the addressed bit is set (JC, JB, JBC) or if the addressed bit is not set (JNC, JNB). JBC executes the jump if the addressed bit is set, and also clears the bit. Thus a flag can be tested and cleared in one operation. All the PSW bits are directly addressable, so the Parity bit, or the general purpose flags, for example, is also available to the bit-test instructions.

Table 4-8. **Boolean Instructions**

MNEMONIC	OPERATION	EXECUTION TIME (mc)
ANL C,bit	C = C.AND.bit	1
ANL C,/bit	C = C.AND..NOT.bit	1
ORL C,bit	C = C.OR.bit	1
ORL C,/bit	C = C.OR..NOT.bit	1
MOV C,bit	C = bit	1
MOV bit,C	bit = C	1
CLR C	C = 0	1
CLR bit	bit = 0	1
SETB C	C = 1	1
SETB bit	bit = 1	1
CPL C	C = .NOT.C	1
CPL bit	bit = .NOT.bit	1
JC rel	Jump if C = 1	2
JNC rel	Jump if C = 0	2
JB bit,rel	Jump if bit = 1	2
JNB bit,rel	Jump if bit = 0	2
JBC bit,rel	Jump if bit = 1; CLR bit	2

Relative Offset

The destination address for these jumps is specified to the assembler by a label or by an actual address in Program memory. However, the destination address assembles to a relative offset byte. This is a signed (two's complement) offset byte which is added to the PC in two's complement arithmetic if the jump is executed. The range of the jump is therefore –128 to +127 Program Memory bytes relative to the first byte following the instruction.

4.4.7 Jump Instructions

Table 4-9 shows the list of unconditional jumps and the execution time associated.

The table lists SJMP, LJMP, and AJMP, which differ in the format of the destination address. JMP is a generic mnemonic which can be used if the programmer does not care which way the jump is encoded.

The SJMP instruction encodes the destination address as a relative offset, as described above. The instruction is 2 bytes long, consisting of the opcode and the relative offset byte. The jump distance is limited to a range of –128 to +127 bytes relative to the instruction following the SJMP.

The LJMP instruction encodes the destination address as a 16-bit constant. The instruction is 3 bytes long, consisting of the opcode and two address bytes. The destination address can be anywhere in the 64k Program Memory space.

The AJMP instruction encodes the destination address as an 11-bit constant. The instruction is 2 bytes long, consisting of the opcode, which itself contains 3 of the 11 address bits, followed by another byte containing the low 8 bits of the destination address. When the instruction is executed, these 11 bits replace the low 11 bits in the PC. The high 5 bits stay the same. Hence the destination has to be within the same 2k block as the instruction following the AJMP.

The JMP @A+DPTR instruction supports case jumps. The destination address is computed at execution time as the sum of the 16-bit DPTR register and the Accumulator. Typically, DPTR is set up with the address of a jump table.

Table 4-9 shows two “CALL addr” instructions LCALL and ACALL, which differ in the format in which the subroutine address is given to the CPU. CALL is a generic mnemonic which can be used if the programmer does not care which way the address is encoded.

The LCALL instruction uses the 16-bit address format, and the subroutine can be anywhere in the 64k Program Memory space.

The ACALL instruction uses the 11-bit format, and the subroutine must be in the same 2k block as the instruction following the ACALL.

Subroutines should end with a RET instruction, which returns execution to the instruction following the CALL.

RETI is used to return from an interrupt service routine. The only difference between RET and RETI is that RETI tells the interrupt control system that the interrupt in progress is done. If there is no interrupt in progress at the time RETI is executed, then the RETI is functionally identical to RET.

Table 4-10 shows the list of conditional jumps available to the microcontroller user. All of these jumps specify the destination address by the relative offset method, and so are limited to a jump distance of –128 to +127 bytes from the instruction following the conditional jump instruction.

There is no Zero bit in the PSW. The JZ and JNZ instructions test the Accumulator data for that condition.

The DJNZ instruction (Decrement and Jump if Not Zero) is for loop control.

The CJNE instruction (Compare and Jump if Not Equal) can also be used for loop control. Two bytes are specified in the operand field of the instruction. The jump is executed only if the two bytes are not equal. Another application of this instruction is in “greater than, less than” comparisons. The two bytes in the operand field are taken as unsigned integers. If the first is less than the second, then the Carry bit is set (1). If the first is greater than or equal to the second, then the Carry bit is cleared.

Table 4-9. **Unconditional Jump instructions**

MNEMONIC	OPERATION	EXECUTION TIME (mc)
(S)JMP addr	Jump to addr	2
(L)JMP addr	Jump to addr	2
(A)JMP addr	Jump to addr	2
JMP @A+DPTR	Jump to A + DPTR	2
(A,L)CALL addr	Call subroutine at addr	2
RET	Return from subroutine	2
RETI	Return from interrupt	2
NOP	No operation	1

Table 4-10. **Conditional Jump instructions**

MNEMONIC	OPERATION	ADDRESSING MODES				EXECUTION
		DIR	IND	REG	IMM	TIME(mc)
JZ rel	Jump if A = 0					2
JNZ rel	Jump if A /= 0					2
DJNZ <byte>,rel	Decrement and jump if not zero	X		X		2
CJNE A,<byte>,rel	Jump if A /= <byte>	X			X	2
CJNE <byte>,#data,rel	Jump if <byte> /= #data		X	X		2

4.5 CPU Timing

Execution time is divided into machine cycles. A machine cycle consists of a sequence of 3 states, numbered S1 through S3. Each state time lasts for one clock period. Thus a machine cycle takes 3 clock periods (or near 0.27µs if the clock frequency is 11.059.200Hz).

Figure 4-10 shows fetch/execute sequences for various kinds of instructions; while an instruction is being executed code bytes for the next instruction are fetched. Normally three program fetches are generated during each machine cycle, even if the next instruction to be executed doesn't require it. If the instruction doesn't need the three code bytes, the CPU simply ignores the extra fetches, and the Program Fetch Counter is not incremented.

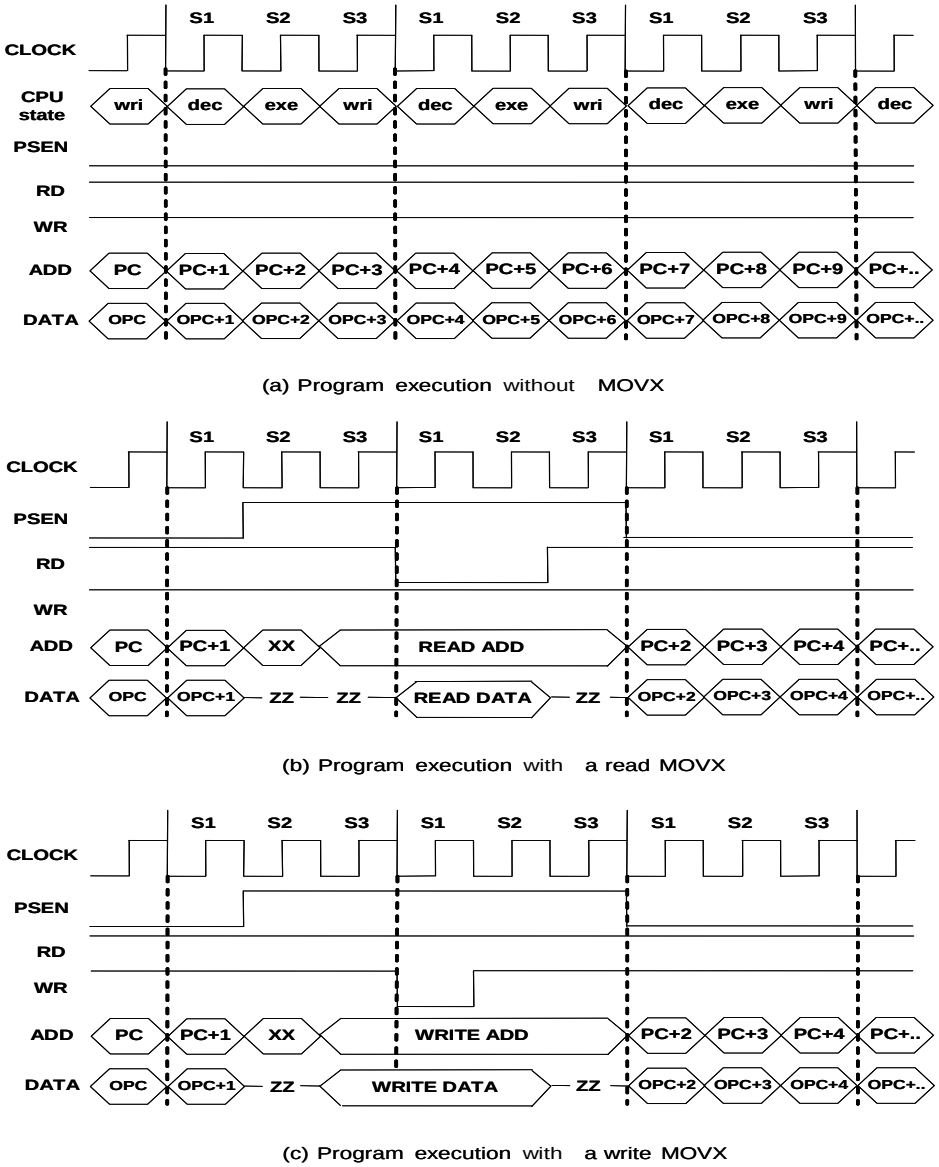
All the instructions, except MOVX, have the same fetch/execute sequence.

The MOVX instructions take two machine cycles to execute. During a MOVX instruction only one fetch is generated, S1 in machine cycle 1. This is the only time program fetches are skipped. The fetch/execute sequence for MOVX instructions is shown in Figure 4-10b and Figure 4-10c.

The fetch/execute sequences are the same whether the Program Memory is internal or external to the chip. Execution times do not depend on whether the Program Memory is internal or external.

Figure 4-10 shows the signals and timing involved in program fetches when the Program Memory is external. If Program Memory is external, then the Program Memory read strobe PSEN is normally activated ('0') three times per machine cycle, as shown in Figure 4-10a. If an access to external Data Memory occurs, as shown in Figure 4-10b and 15c, five PSENs are skipped, because the address and data bus are being used for the Data Memory access.

Figure 4-10. Bus timing in 8051C3A



(a) Program execution without MOVX

(b) Program execution with a read MOVX

(c) Program execution with a write MOVX

4.5.1 Reset

ATPL00B can be reset in different ways, listed and explained below

- **External reset.** The external reset input signal is RSTA which is an asynchronous reset. Every time RSTA signal is asserted, the boot loader is triggered (thus, microcontroller is halted, target program from the serial flash is uploaded to the internal SRAM, and then microcontroller is released to begin program execution). In power-on, D_INIT must be set before RSTA in order to ensure proper system start up.
- **Software reset →** AUX1(5) = AUX1.SRST. This field in AUX1 SFR forces the microcontroller to start program execution from PC=0, and all the SFRs are initialized to their reset default values. This reset doesn't affect the boot loader reload.
- **Software reset →** CONF(2) = CONF.RLD. This field is directly connected to boot loader reload input. Every time the microcontroller sets this field, the boot loader is triggered.

- **Watchdog reset** → /EWDG signal and CONF(1)=CONF.RLT3.
  - CONF.RLT3 = ‘0’ : After a watchdog timeout, microcontroller starts execution from PC=0, but boot loader is not triggered.
  - CONF.RLT3 = ‘1’ **:** After a watchdog timeout, if field RLT3 in CONF SFR is ‘1', boot loader is triggered.

The microcontroller keeps track of the start-up point in the AUX2 register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
AUX2	--	--	--	--	--	ST2	ST1	ST0

- --: Reserved bits
- ST(2:0): Start-up flags
  - “000”:Asynchronous external reset
  - “001”:Software reset by writing AUX1(5)=’1’
  - “010”:Watchdog T3 timeout
  - “100”:Asynchronous external reset

SFR values after a reset are shown in Table 4-11. The internal RAM is not affected. Port registers are initialized to FFH to keep port pins in pull-up state.

Table 4-11. **SFRs Reset values**

SFR	Address	Reset Value
P0	80H	11111111
SP	81H	00000111
DPL	82H	00000000
DPH	83H	00000000
PCON	87H	00000000
TCON	88H	00000000
TMOD	89H	00000000
TL0	8AH	00000000
TL1	8BH	00000000
TH0	8CH	00000000
TH1	8DH	00000000
TPR	8EH	00111111
T1NC	8FH	00000000
P1	90H	11111111
IP2HL	91H	xxxx0000
SCON2	94H	00000000
SBUF2	95H	00000000
T1NP	96H	00000000
SCON	98H	00000000
SBUF	99H	00000000
SCON1	9AH	00000000

SFR	Address	Reset Value
SBUF1	9BH	00000000
P3M	9CH	00000000
P4M	9DH	00000000
P5M	9EH	00000000
P2	A0H	11111111
IE2	A1H	xxxxxx00
AUX1	A2H	00000000
AUX2	A3H	xxxxxbbb
IE	A8H	00000000
P3	B0H	11111111
IPH	B7H	00000000
IP	B8H	00000000
P4	C0H	11111111
P5	C4H	11111111
T2CON	C8H	00000000
T2MOD	C9H	xx001100
RCAP2L	CAH	00000000
RCAP2H	CBH	00000000
TL2	CCH	00000000
TH2	CDH	00000000
PSW	D0H	00000000

SFR	Address	Reset Value
SPDAT10	D8H	00000000
SPDAT11	D9H	00000000
SPDAT12	DAH	00000000
SPDAT13	DBH	00000000
SPDAT14	DCH	00000000
SPDAT15	DDH	00000000
SPDAT16	DEH	00000000
SPDAT17	DFH	00000000
ACC	E0H	00000000
SPSTAT	E1H	000xxxxx
SPCTL	E2H	00000100
SPDAT	E3H	00000000
SPSTAT1	E4H	000xx000
SPCTL1	E5H	00000100
B	F0H	00000000
CONF	F1H	00000000
T3	FFH	00000000

4.5.2 Power Saving Modes

ADD8051C3A has two power-saving modes, Idle and Power Down.

In the Idle mode (IDL = 1), the CPU is stopped but the peripheral circuits continue running.

In Power Down (PD = 1), the CPU and the peripheral circuits are stopped, except external interrupts hardware that can be configured to run when in power down mode.

The Idle and Power Down Modes are activated by setting bits in Special Function Register PCON. If 1s are written to PD and IDL at the same time, PD takes precedence. When watchdog is enabled power down modes are automatically disabled.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
PCON	SMOD	--	--	WLE	GF1	GF0	PD	IDL

- SMOD: Double baud rate bit when timer 1 is used as time generator and serial port is in modes 1, 2 or 3
- --: Reserved bits
- WLE: Watchdog load enable. It must be set by software to enable T3 reload. It is reset by hardware after 13 machine cycles
- GF1: General purpose flag bit, user programmable
- GF0: General purpose flag bit, user programmable
- PD: Sets power-down mode
- IDL: Sets idle mode

4.5.3 Idle Mode

Setting PCON(0), CPU is stopped but peripheral circuits (Interrupts, Timers, Standard Serial Interface and Serial Peripheral Interface functions) continue running. The CPU status is entirely preserved; Stack Pointer, Program Counter, Program Status Word, Accumulator, and all other registers maintain their data during Idle. The port pins hold the logical states they had at the time Idle was activated. PSEN holds at logic high level.

There are two ways to terminate the Idle mode.

- Activation of any enabled interrupt will cause PCON(0) to be cleared by hardware, terminating the Idle mode. The interrupt will be served, and following RETI, the next instruction to be executed will be the one following the instruction that put the device into Idle.
- Reset signal. The signal at the RSTA pin redefines all the SFR values to their reset values and then the IDL bit is cleared. At this time, and with three clock cycle delay, the CPU restart program execution with program counter at initial value 0x0000, RAM content remains unchanged.

4.5.4 Power-Down Mode

An instruction that sets PCON(1) causes that to be the last instruction executed before going into the Power Down mode. In the Power Down mode all functions are stopped, except external interrupts hardware. The contents of the on-chip RAM and Special Function Registers are maintained. The port pins output the values held by their respective SFRs. The PSEN output is held low.

There are two ways to exit Power Down mode.

- Hardware reset. Hardware reset redefines all the SFRs.
- External interrupt. External interrupts must be configured for low level activation.When they are enabled a low level at their inputs will cause a microcontroller restart and a jump to the corresponding interruption routine. As in idle mode, the interrupt will be served, and following RETI, the next instruction to be executed will be the one following the instruction that put the device into power down.

4.6 Interrupts

The microcontroller has 10 user interrupt sources:

- 2 external interrupts.
- 3 timer interrupts.
- 3 serial port interrupts.
- 2 standard serial peripheral interrupts.

Interrupt priorities are user selectable to a priority level ranging from “000” to “011” (higher value correspond to higher priority level).

4.6.1 Interrupt Enabling

Each interrupt source can be individually enabled or disabled by setting or clearing a bit in the SFRs named IE and IE2 (Interrupt Enable). The IE register also contains a global disable bit, which can be cleared to disable all interrupts at once.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
IE	EA	ET2	ESPI	ES	ET1	EX1	ET0	EX0

- EA: ‘1’ enables all interrupts; ‘0’ disables all interrupts
- ET2: ‘1’ enables timer 2 interrupt; ‘0’ disables timer 2 interrupt
- ESPI: ‘1’ enables serial peripheral interface0 interrupt; ‘0’ disables serial peripheral interface0 interrupt
- ES: ‘1’ enables serial port 0 interrupt; ‘0’ disables serial port0 interrupt
- ET1: ‘1’ enables timer 1 interrupt; ‘0’ disables timer 1 interrupt
- EX1: ‘1’ enables external interrupt 1; ‘0’ disables external interrupt 1
- ET0: ‘1’ enables timer 0 interrupt; ‘0’ disables timer 0 interrupt
- EX0: ‘1’ enables external interrupt 0; ‘0’ disables external interrupt 0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
IE2	--	--	--	--	ESPI1	ES2	--	ES1

- --: Reserved bit
- ESPI1: ‘1’ enables serial peripheral interface1 interrupt; ‘0’ disables serial peripheral interface1 interrupt.
- ES2: ‘1’ enables serial port 2 interrupt; ‘0’ disables serial port2 interrupt
- ES1: ‘1’ enables serial port 1 interrupt; ‘0’ disables serial port1 interrupt

4.6.2 Interrupt Priorities

Each interrupt source can also be individually programmed to four priority levels by setting or clearing two bits in the SFRs named IP, IPH and IP2HL (Interrupt Priority). Figure 4-11 shows how the IE, IE2, IP, IPH and IP2HL registers and the polling sequence work to determine the interrupt to be served. The IP, IPH, IP2HL and IE2 registers contain a number of unimplemented bits, these bits are reserved and user software should not write to these positions.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
IP	--	PT2	PSPI	PS0	PT1	PX1	PT0	PX0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
IPH	--	PT2H	PSPIH	PS0H	PT1H	PX1H	PT0H	PX0H

- -- : Reserved bit
- [IPH(6),IP(6)] : PT2 defines timer 2 interrupt priority level
- [IPH(5),IP(5)] : PSPI defines serial peripheral interface SPI0 interrupt priority level
- [IPH(4),IP(4)] : PS0 defines serial port 0 interrupt priority level
- [IPH(3),IP(3)] : PT1 defines timer 1 interrupt priority level
- [IPH(2),IP(2)] : PX1 defines external interrupt 1 priority level
- [IPH(1),IP(1)] : PT0 defines timer 0 interrupt priority level
- [IPH(0),IP(0)] : PX0 defines external interrupt 0 priority level

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
IP2HL	PSPI1_1	PSPI1_0	PS2_1	PS2_0	--	--	PS1_1	PS1_0

- IP2HL(7:6) : PSPI1 defines serial peripheral port 1 SPI1 priority level
- IP2HL(5:4) : PS2 defines serial port 2 priority level
- -- : Reserved bit
- IP2HL(1:0) : PS1 defines serial port 1 priority level

A low-priority interrupt can only be interrupted by a higher priority interrupt (but neither by another low-priority nor an equal-priority interrupt).

If two interrupt requests of different priority levels are received simultaneously, the request of higher priority is served. If interrupt requests of the same priority level are received simultaneously, an internal polling sequence determines which request is served. Thus within each priority level there is a second priority structure determined by the polling sequence as follows:

0.	IE0	(highest priority)
1.	TF0	↑
2.	IE1	
3.	TF1	
4.	RI+TI	
5.	SPI0	
6.	TF2+EXF2	
7.	RI1+TI1	
9.	RI2+TI2	
10.	SPI1	(lowest priority)

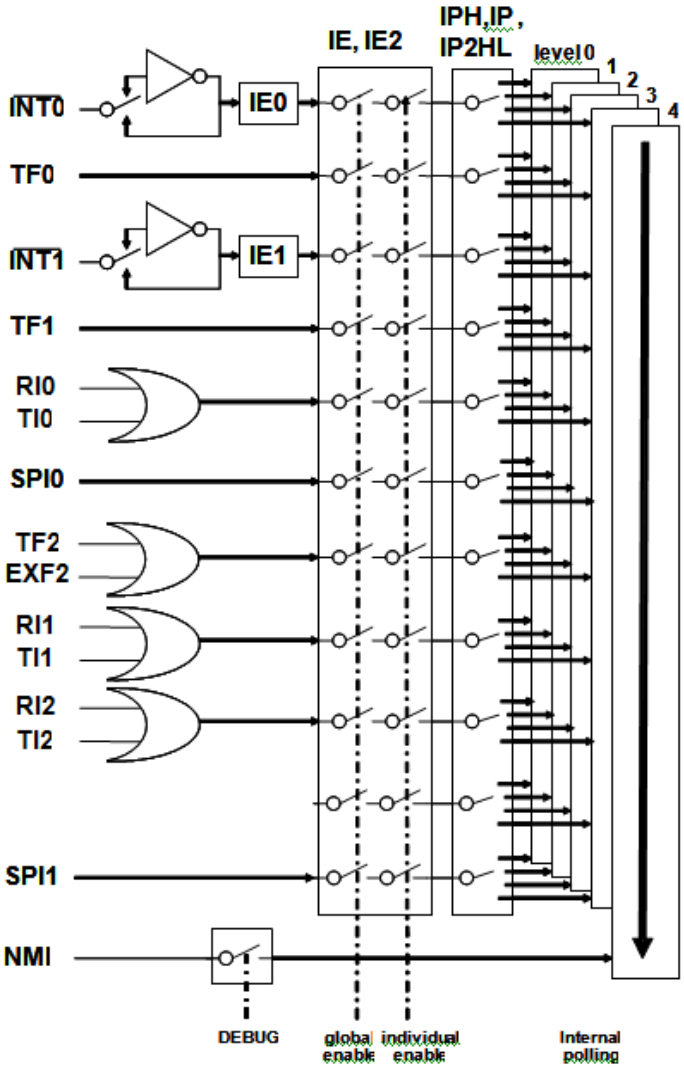
When in operation mode, all the interrupt flags are latched into the interrupt control system during state 2 of every machine cycle. The samples are polled during the following machine cycle. If the flag for an enabled interrupt is found to be set ('1'), the interrupt system generates an LCALL to the appropriate location in Program Memory, unless some other condition blocks the interrupt. Several conditions can block an interrupt, for example when an interrupt of equal or higher priority level is already in progress.

The hardware-generated LCALL causes the value in the Program Counter to be pushed into the stack, and reloads the PC with the beginning address of the service routine. As previously noted the service routine for each interrupt begins at a fixed location.

Only the Program Counter is automatically pushed onto the stack, not the PSW or any other register.

Having only the PC automatically saved it is a programmer task to save other registers if necessary

Figure 4-11. Interrupt Priorities diagram



4.6.3 Interrupt Handling

External Interrupts INT0 and INT1 can each be either level-activated or transition-activated, depending on bits IT0 and IT1 in Register TCON. When external interrupts are transition-activated the INTx pin must show a high level in one cycle and a low level in the next cycle to generate the interrupt, so that an input high or low should hold for at least 1 machine cycle to ensure sampling. The flags that actually generate these interrupts are bits IE0 and IE1 in TCON. The flags are cleared by hardware when the service routine is vectored only if the interrupts are transition-activated. When the interrupts are level-activated the interrupt flags are controlled by the external source, so that the external source has to deactivate the request before the interrupt service routine is completed, or else another interrupt will be generated. The response time is always more than 3 machine cycles and less than 7 machine cycles depending on program execution and interrupt status (see Figure 4-12).

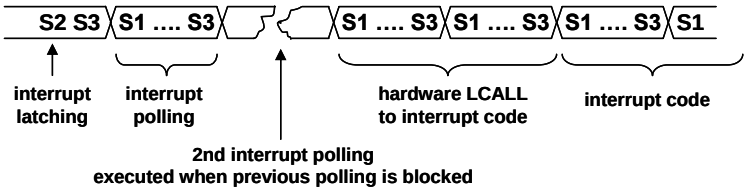
Timer 0 and Timer 1 Interrupts are generated by TF0 and TF1, which are set by a rollover. The flags are cleared by the on-chip hardware when the service routines are vectored.

Timer 2 interrupts (TF2 and EXF2) are user configurable, and depending on configuration flags will be cleared by hardware or by user software.

Serial Ports Interrupts are generated by the logical OR of RI and TI. They must be cleared by software.

SPI interrupts must be cleared by software.

Figure 4-12. Interrupt Handling



All of the bits that generate interrupts can be set or cleared by software. That is, interrupts can be generated or pending interrupts can be canceled by software. Each of these interrupt sources can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE. IE also contains a global disable bit, EA, which disables all interrupts at once.

The interrupt flags are sampled at every machine cycle. The samples are polled during the following machine cycle. If one of the flags was in a set condition at the preceding cycle, the polling cycle will find it and the interrupt system will generate an LCALL to the appropriate service routine, provided this hardware-generated LCALL is not blocked by any of the following conditions:

- Condition 1. An interrupt of equal or higher priority level is already in progress.
- Condition 2. The current (polling) cycle is not the final cycle in the execution of the instruction in progress.
- Condition 3. The instruction in progress is RETI or any write to the IE or IP registers.

Any of these three conditions will block the generation of the LCALL to the interrupt service routine. Condition 2 ensures that the instruction in progress will be completed before vectoring to any service routine. Condition 3 ensures that if the instruction in progress is RETI or any access to IE or IP, then at least one more instruction will be executed before any interrupt is vectored to.

The polling cycle is repeated with each machine cycle, and the values polled are the values that were present at the previous machine cycle. The polling cycle/LCALL sequence is illustrated in Figure 4-12.

The hardware-generated LCALL executes a program jump to fixed program entries as shown below:

Table 4-12. **Fixed Program entries**

Source	Vector Address
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
RI0+TI0	0023H
SPI0	002BH
TF2+EXF2	0033H
RI1+TI1	003BH
RI2+TI2	004BH
SPI1	0053H

Execution proceeds from that location until the RETI instruction is encountered then the interrupted program continues from where it left off.

4.7 I/O Ports

The ports P1, P3, P4 and P5 in the ADD8051C3A are bidirectional. Each port consists of a register (Special Function Registers P1, P3, P4, P5), an output driver, and an input buffer.

Pins in ports 3, 4 and 5 are multifunctional. They are port pins and also serve as input or output for the microcontroller peripherals:

Table 4-13. **P3, P4, P5 ports Alternate Functions**

Pin	Alternate Function
P3.0	RxD (serial port 0 input)
P3.1	TxD (serial port 0 output)
P3.2	INT0 (external interrupt 0)
P3.3	INT1 (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	WR(external data memory write strobe)
P3.7	RD (external data memory read strobe)
P4.0	RxD (serial port 2 input)
P4.1	TxD (serial port 2 output)
P4.2	SS1 (SPI1 slave select input)
P4.3	SPICLK1 (SPI1 clock input/output)
P4.4	MOSI1 (SPI1 master out / slave in data)
P4.5	MISO1 (SPI1 master in / slave out data
P4.6	T2 (timer 2 input/output)
P4.7	T2EX (timer 2 external input)
P5.0	SS (SPI0 slave select input)
P5.1	SPICLK (SPI0 clock input/output)
P5.2	MOSI (SPI0 master output / slave input data)
P5.3	MISO (SPI0 master input / slave output data)
P5.4	RxD (serial port 1 input)
P5.5	TxD (serial port 1 output)

Note: To use an Alternate Function the corresponding bit (or bits) in the SFR must contain a 1.

4.7.1 I/O Configurations

Figure 4-13 and Figure 4-14 show a functional diagram of bit register and I/O buffer in each of the four ports. The level of the port pin is placed on the internal bus and instructions can read the port pin value or the SFR register value.

If a bit in a register with AOF (Alternate Output Function) contains a 1, then the output level is controlled by the signal labeled “Alternate Output function”.

Figure 4-13. Pins with AOF structure

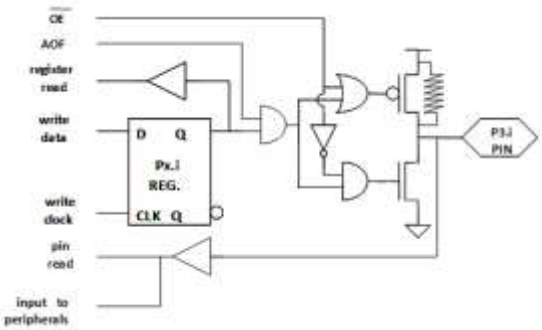
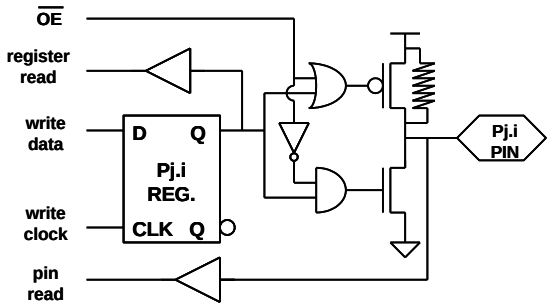


Figure 4-14. Pins without AOF structure



Pseudo-bidirectional mode

Ports 1, 3, 4 and 5 have internal pull-ups. Each I/O line can be independently used as an input or an output. To be used as an input, the port bit register must contain a ‘1’; When an SFR port bit is set to ‘1’ the PMOS port driver is turned on one clock cycle and then is turned off. Then, the pin is pulled high by a weak internal pull-up, and can be pulled low by an external source. In the ADD8051C3A included in ATPL00B, the pull-up consists of 33Kohm resistors.

P1, P3, P4 and P5 registers are set to FFH by default (configured as inputs).

Push-Pull mode

Ports 3, 4 and 5 can be configured as Push-Pull output ports using P3M, P4M and P5M SFRs respectively.

Setting a bit in these SFRs to `1’ configures the corresponding port as Push-Pull mode.

Setting a bit in these SFRs to ‘0’ configures the corresponding port as Pseudo-bidirectional.

P3M, P4M and P5M are set to 00H by default (configured in Pseudo-bidirectional mode).

4.7.2 Read-Modify-Write Feature

Some instructions that read a port read the register and others read the pin. Only the read-modify-write instruction read the register. The instructions that read the register rather than the pin are the ones that read a value, possibly change it, and then rewrite it to the register. When the destination operand is a port, or a port bit, these instructions read the register rather than the pin: ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ, MOV PX.Y,C, CLR PX.Y, SET PX.Y. The last three instructions are bit modify instructions. They read the port byte, all 8 bits, modify the target bit, then the modified byte is written to the register.

The-modify-write instructions are directed to the register to avoid a possible misinterpretation of the voltage level at the pin. When a port register is set to ‘1’ to using the pin as input an external source can be pulling the pin to ‘0’, if we need to know the configuration of the port we need to read the register and to know the state of the external source we need to read the pin.

4.7.3 Accessing External Memories

Accesses to external memory are of two types: accesses to external program code and accesses to external program data. Both share ports ADDRESS and DATA buses. Accesses to external program code use signal PSEN (program store enable) as the read strobe. Accesses to external program data use RD(P3.7) or WR(P3.6) to strobe the memory.

Fetches from external program memory always use a 16-bit logical address, fetches to external data can use 16-bit or 8-bit logical address.

4.8 Debug Mode

Debug mode is active when DBG pin is set to ‘1’. Debug mode is intended to implement software debugging tools.

As stated in Table 2-1, DEBUG digital input is internally connected to DBG signal of the ADD8051C3A microcontroller

When in debug mode, a program can modify its own code. Thus, user **should not** work in debug mode unless explicitly stated otherwise.

5. Timers

The ADD8051C3A has a full set of timers, listed below:

- Two 16-bit configurable Timers/Counters: Timer 0 and Timer 1 that can be configured as timers or event counters.
- Three 8-bit auto reload counters Timers 11, 12 and 14, used as baud rate generators for uarts 0, 1 and 2 respectively (see Standard Serial Interfaces section).
- One 8-bit, with 18-bit preescaler, watchdog timer: Timer 3 (see Watchdog (timer 3) section).
- One 16-bit Timer/Counter Timer 2 with capture reload hardware (see 5.2)

5.1 Timer 0 and Timer 1

Timers 0 and 1 have 2-bit preescalers. These preescalers can be programmed to slow down count rate. The reset value set the preescaler values to the slowest count rate and the real count rate of the timers is the same than the timers in an “8051 legacy” microcontroller (12 clock cycles per machine cycle).

TPR, TMOD and TCON registers described below are involved in timer 0 and timer 1 management.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TPR	--	--	T1P1	T1P0	T01P1	T01P0	T0P1	T0P0

- --: Reserved bit
- T1P(1:0): Rollover value for T1 preescaler when T1 is in modes 0, 1, 2 and 3
- T01P(1:0): Rollover value for TH0 preescaler when T0 is in mode 3
- T0P(1:0): Rollover value for T0 preescaler when T0 is in modes 0, 1, and 2, and for TL0 preescaler when T0 is in mode 3

When T0 or T1 are in the timer function, the registers are incremented every 1, 2, 3 or 4 machine cycles. Since a machine cycle consists of 3 oscillator periods, the count rate is 1/3, 1/6, 1/9 or 1/12 of the oscillator frequency.

In the counter function, the register is incremented in response to a 1-to-0 transition at its corresponding external input pin, T0(P3.4) or T1(P3.5). When in counter function, the external input is sampled once every machine cycle. When the samples show a high in one cycle and a low in the next cycle (1-to-0 transition), the count is incremented. The register value is updated during the cycle following the one in which the transition was detected, the maximum count rate is 1/2 of the machine cycle frequency.

Either the timer or counter function is selected by control bit C/T in the Special Function Register TMOD where T0 and T1 have four operating modes, which are selected by bit-pairs (M1, M0) in TMOD.

	TIMER 1				TIMER 0			
Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TMOD	GATE	C/T	M1	M0	GATE	C/T	M1	M0

- GATE:
  - ‘1’: Enables gate control. Device count is enabled only when both ‘INTi’ pin and “TCON.TRi” are high.
  - ‘0’: Disables gate control. Device count is enabled when “TCON.TRi” is high.
- C/T:
  - ‘1’: Counter operation mode. Input from Ti input pin

‘0’: Timer operation mode. Input from internal clock system

- M(1:0): Operation modes

  “00”: Mode 0: 8048 13bit timer, TLi used as 5 bit preescaler

  “01”: Mode 1: 16 bit timer/counter composed by concatenated THi and TLi

  “10”: Mode 2: 8 bit with auto reload (TLi <= THi) timer/counter

  “11”: Mode 3: timer 1 stopped

  Timer 0 runs as two 8-bit independent timer/counters:

  TL0 is a timer/counter controlled by timer 0 control bits

  TH0 is a timer controlled by timer 1 control bits

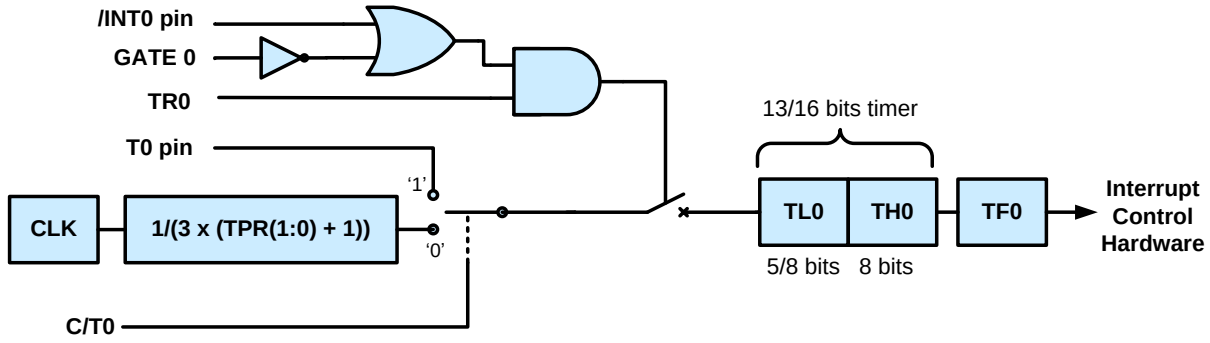
Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TCON	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

- TF1: Timer 1 overflow flag, set by hardware overflow and cleared by software or by hardware when interrupt routine is vectored
- TR1: Timer 1 run control bit, when set turns on timer/counter 1
- TF0: Timer 0 overflow flag, set by hardware overflow and cleared by software or by hardware when interrupt routine is vectored
- TR0: Timer 0 run control bit, when set turns on timer/counter 0
- IE1: Interrupt 1 edge flag, set by hardware when an external interrupt edge is detected and cleared by software or by hardware when interrupt routine is vectored
- IT1: Interrupt 1 type control bit. “’1’/’0’” to specify “falling edge/low level” trigger for external interruption 1
- IE0: Interrupt 0 edge flag, set by hardware when an external interrupt edge is detected and cleared by software or by hardware when interrupt routine is vectored
- IT0: Interrupt 0 type control bit. “’1’/’0’” to specify “falling edge/low level” trigger for external interruption 0

5.1.1 Timer Mode 0

The timer (0 or 1) into Mode 0 looks like a 13-bit timer: an 8-bit counter (all 8 bits of THx) with a divide-by-32 preescaler (the lower 5 bits of TLx), see Figure 5-1. As the count rolls over from all 1s to all 0s, it sets the Timer interrupt flag TFi. The counter input is enabled to the Timer when TRi=1 and either GATE=0 or INT1=1. When GATE=1 the Timer can be controlled by external input INTx, to facilitate pulse width measurements. TRx is a control bit in the Special Function Register TCON.

Figure 5-1. Timer 0 (or Timer 1) in modes 0,1



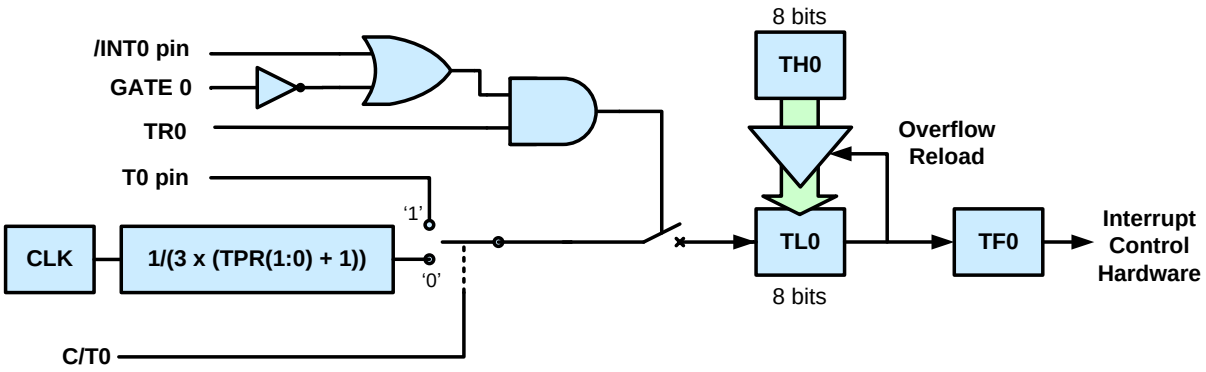
5.1.2 Timer Mode 1

In Mode 1 the timer (0 or 1) runs in the same way than Mode 0 but they are 16 bits-width and TLx is not used as a preescaler.

5.1.3 Timer Mode 2

Mode 2 configures the timer (0 or 1) register as an 8-bit Counter (TLx) with automatic reload of THx value. THx value is preset by software.

Figure 5-2. Timer 0 (or Timer 1) in mode 2



5.1.4 Timer Mode 3

In Mode 3 Timer 0 registers TL0 and TH0 are configured as two separate 8-bit counters, TL0 and TH0.

TL0 uses the Timer 0 control bits and can be configured either in timer or in counter mode.

TH0 is locked into a timer function (counting machine cycles) and is controlled by Timer 1 control bits TR1 and TF1.

TH0 controls the Timer 1 interrupt.

Timer 1 in Mode 3 holds its count. With Timer 0 in Mode 3, the number of timer/counters in the 80C51 is increased to three. When Timer 0 is in Mode 3, Timer 1 can be used by the serial port as a baud rate generator, or in any application not requiring an interrupt.

5.2 Timer 2

Timer 2 is a 16-bit timer/counter. The count is maintained by two eight-bit timer registers, TH2 and TL2, connected in cascade. T2MOD and T2CON registers described below control the operation of timer 2.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
T2MOD	--	--	TF2SC	EXF2SC	T2PR1	T2PR0	T2OE	DCEN

- --: Reserved bit
- TF2SC: Timer 2 overflow flag software control. When set to ‘1’, TF2 flag must be cleared by software. When cleared to ‘0’, TF2 flag is cleared by hardware when interrupt routine is vectored
- EXF2SC: External Interrupt 2 flag software control. When set to ‘1’, EXF2 flag must be cleared by software. When cleared to ‘0’, EXF2 flag is cleared by hardware when interrupt routine is vectored
- T2PR(1:0): Timer 2 preescaler
- T2OE: Timer 2 output enable bit. In the timer 2 clock-out mode connects the programmable clock input to the external pin T2
- DCEN: Down count enable bit, configures timer 2 as an up/down counter

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
T2CON	TF2	EXF2	SSI1	SSI0	EXEN2	TR2	C/T2	CP/RL2

- TF2: Timer 2 overflow flag, Set by timer 2 overflow. It must be cleared by software. TF2 is not set when timer 2 is in baud rate generator mode
- EXF2: Timer 2 external flag. If EXEN2=’1’ capture or reload caused by a negative transition on T2EX sets EXF2. EXF2 does not cause an interrupt in up/down counter mode (DCEN=’1’)
- SSI(1:0): Select T2 as baud rate generator for serial port
  - “00”: none
  - “01”: USART0
  - “10”: USART1
  - “11”: USART2
- EXEN2: Timer 2 external enable bit. Setting EXEN2 causes a capture or reload to occur as a result of a negative transition on T2EX unless timer 2 is being used as baud rate generator. Clearing EXEN2 causes timer 2 to ignore events at T2EX
- TR2: Timer 2 run control bit. Setting this bit starts the timer
- C/T2: Timer 2 counter/timer select. When set it selects counter operation then timer 2 counts negative transitions on system clock
- CP/RL2: When set to ‘1’, captures occur on negative transitions at T2EX if EXEN=’1’; When cleared to ‘0’, auto reload occurs on timer 2 overflow or negative transitions at T2EX if EXEN2=`1`. If RCLK=’1’ or TCLK=’1’ the CP/RL2 bit is ignored

Timer 2 provides the following operating modes: capture mode, auto-reload mode, baud rate generator mode, and programmable clock-out mode. Select the operating mode with T2MOD and T2CON register bits as shown in Table 5-1. Auto-reload is the default mode. Setting T2CON(5:4) selects the baud rate generator mode.

Table 5-1. **Timer 2 operating modes**

Mode	T2CON(5) OR T2CON(4)	CP/RL2	T2OE
Auto-reload	0	0	0
Programmable clock-out	0	0	1
Capture	0	1	0
stopped	0	1	1
Baud Rate generator	1	X	X

Timer 2 operation is similar to timer 0 and timer 1. C/T2 selects between timer operation mode (internal clock input) or counter operation mode (external pin T2 as the time register input). Setting TR2 allows TL2 to be incremented by the selected input.

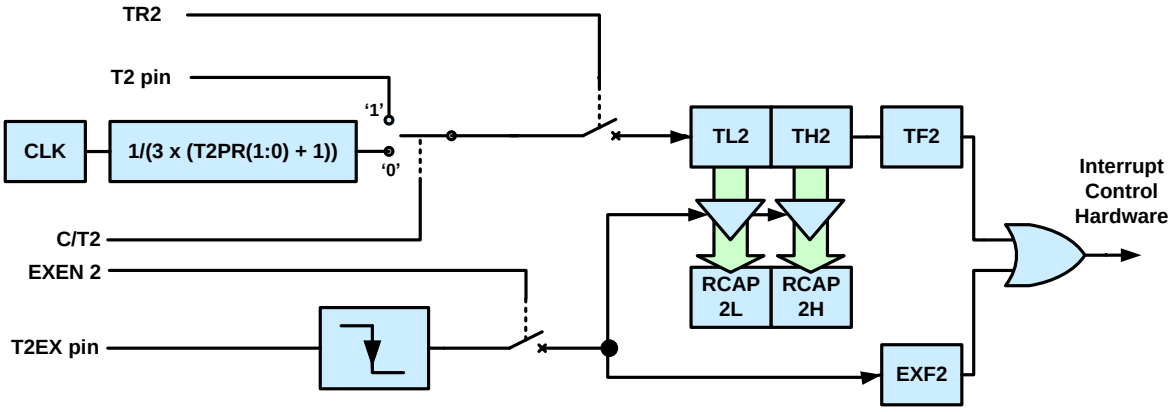
The timer 2 related interrupt flags (TF2, EXF2) can be configured to be cleared by hardware when the interrupt routine is vectored to, or by user software using TF2SC and EXF2SC.

The operating modes are described in the following paragraphs.

5.2.2 Capture mode

In the capture mode, timer 2 operates as a 16-bit timer or counter. An overflow condition sets bit TF2, which can be used to request an interrupt. Setting the external enable bit EXEN2 allows the RCAP2H and RCAP2L registers to capture the current value in timer registers TH2 and TL2 in response to a 1-to-0 transition at external input T2EX. The transition at T2EX also sets bit EXF2 in T2CON. The EXF2 bit, like TF2, can generate an interrupt.

Figure 5-3. Timer 2 capture mode



5.2.3 Auto-Reload mode

The auto-reload mode configures timer 2 as a 16-bit timer or event counter with automatic reload. The timer operates as an up counter or as an up/down counter, as determined by the down counter enable bit (DCEN). When reset occurs, DCEN is cleared, so in the auto-reload mode, timer 2 in configured as an up counter by default.

- **Up Counter Operation**

  When DCEN = 0, timer 2 operates as an up counter.

  The external enable bit EXEN2 in the T2CON register provides two options. If EXEN2 = 0, timer 2 counts up to FFFFH and sets the TF2 overflow flag. The overflow condition loads the 16-bit value in the reload/capture registers (RCAP2H, RCAP2L) into the timer registers (TH2, TL2). The values in RCAP2H and RCAP2L are preset by software.

  If EXEN2 = 1, the timer registers are reloaded by either a timer overflow or a high-to-low transition at external input T2EX. This transition also sets the EXF2 bit in the T2CON register.

  Either TF2 or EXF2 bit can generate a timer 2 interrupt request.

- **Up/Down Counter Operation**

  When DCEN = 1, timer 2 operates as an up/down counter.

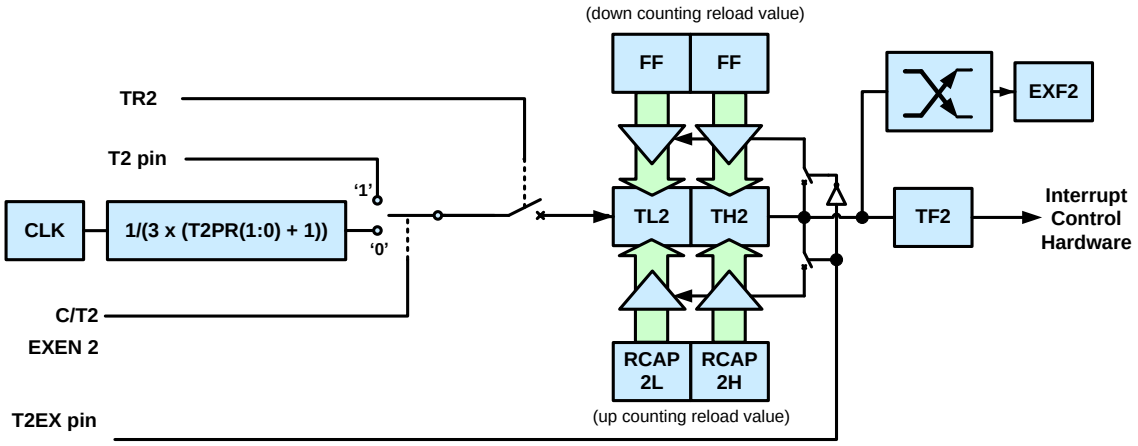
  External pin T2EX controls the direction of the count.

  When T2EX is high, timer 2 counts up. The timer overflow occurs at FFFFH which sets the timer 2 overflow flag (TF2) and generates an interrupt request. The overflow also causes the 16-bit value in RCAP2H and RCAP2L to be loaded into the timer registers TH2 and TL2.

  When T2EX is low, timer 2 counts down. Timer underflow occurs when the count in the timer registers (TH2, TL2) equals the value stored in RCAP2H and RCAP2L. The underflow sets the TF2 bit and reloads FFFFH into the timer registers.

  The EXF2 bit toggles when timer 2 overflows or underflows changing the direction of the count. When timer 2 operates as an up/down counter, EXF2 does not generate an interrupt. This bit can be used to provide 17-bit resolution.

Table 5-2. **Timer 2 Auto-reload mode**



5.2.4 Clock-Out mode

In the clock-out mode, timer 2 operates as a 50%-duty-cycle, variable-frequency clock. The input clock increments TL2 at frequency FOSC/2 (where FOSC=11.059.200Hz). The timer repeatedly counts to overflow from a preloaded value. At overflow, the contents of the RCAP2H and RCAP2L registers are loaded into TH2/TL2. In this mode, timer 2 overflows do not generate interrupts. The formula gives the clock-out frequency as a function of the system oscillator frequency and the value in the RCAP2H and RCAP2L registers:

For a 11.0592 MHz system clock, timer 2 has a programmable frequency range of 7.03 Hz to 1.8432 MHz. The generated clock signal is brought out to the T2 pin (P4.6).

Timer 2 is programmed for the clock-out mode as follows:

1. Set the T2OE bit in T2MOD. This gates the timer register overflow to the ÷2 counter.
2. Clear the C/T2 bit in T2CON to select F_{OSC}/3*(T2PRE(1:0)+1) as the timer input signal. This also gates the output of the ÷2 counter to pin T2.
3. Determine the 16-bit reload value from the formula and enter in the RCAP2H/RCAP2L registers.
4. Enter a 16-bit initial value in timer register TH2/TL2. This can be the same as the reload value, or different, depending on the application.
5. To start the timer, set the TR2 run control bit in T2CON.

Clock-out mode timer 2 operation is similar to timer 2 operating as a baud rate generator. Moreover, it is possible to use timer 2 as a baud rate generator and a clock generator simultaneously. For this configuration, the baud rates and clock frequencies are not independent since both functions use the values in the RCAP2H and RCAP2L registers

The baud rate is expressed by the following formula:

5.2.5 Baud rate Generator mode

This mode configures timer 2 as a baud rate generator for its use with the serial ports. Select this mode by setting the SSI(1:0) bits in T2CON.

Timer 2 may be selected as the baud rate generator for the transmitter and/or receiver in modes 1 and 3. The timer 2 baud rate generator mode is similar to the auto-reload mode. A rollover in the TH2 register reloads registers TH2 and TL2 with the 16-bit value in registers RCAP2H and RCAP2L, which are preset by software.

The timer 2 baud rate is expressed by the following formula:

Where “BAUD2” is the bit in the Special Function Register CONF and RCAP2(H,L) denotes the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

To select timer 2 as baud rate generator for transmision and reception, program the SSI(1:0) bits in the T2CON register as shown in Table 5-1. Setting SSI(1) and/or SSI(0) puts timer 2 into its baud rate generator mode. In this mode, a rollover in the TH2 register does not set the TF2 bit in the T2CON register. Also, a high-to-low transition at the T2EX pin sets the EXF2 bit in the T2CON register but does not cause a reload from (RCAP2H, RCAP2L) to (TH2, TL2).

User shall use the T2EX pin as an additional external interrupt by setting the EXEN2 bit in T2CON.

Turn the timer off (clear the TR2 bit in the T2CON register) before accessing registers TH2, TL2, RCAP2H, and RCAP2L

User shall configure timer 2 as a timer or a counter. In most applications, it is configured for timer operation (i.e., the C/T2 bit is clear in the T2CON register).

Note that timer 2 increments every 1, 2, 3 or 4 state times (3T_{OSC}) when it is in the baud rate generator mode.

When timer 2 is configured as a timer and in baud rate generator mode, do not read or write the TH2 or TL2 registers. The timer is being incremented every state time, and the results of a read or write may not be accurate. In addition, user shall read, but not write to, the RCAP2 registers; a write may overlap a reload and cause write and/or reload errors.

Table 5-3 lists commonly used baud rates and shows how they are generated by timer 2.

Table 5-3. **Timer 2 generated baud rates with a 11.0592MHz oscillator**

Baud rate	T2PRE	RCAP2H	RCAP2L
230400	0	FF	FF
115200	0	FF	FE
57600	0	FF	FC
38400	0	FF	FA
19200	0	FF	F4
9600	0	FF	E8
4800	0	FF	D0
2400	0	FF	A0
1200	0	FF	40
600	0	FE	80
300	0	FD	0
150	0	FA	0
110	0	F7	D2

5.3 Watchdog (timer 3)

The watchdog timer includes a 18-bit preescaler which is set to 0 when T3 is reloaded.

T3 is active only when /EWDG pin is tied to ‘0’.

T3 must be reloaded by software to avoid T3 rollover and program restart.

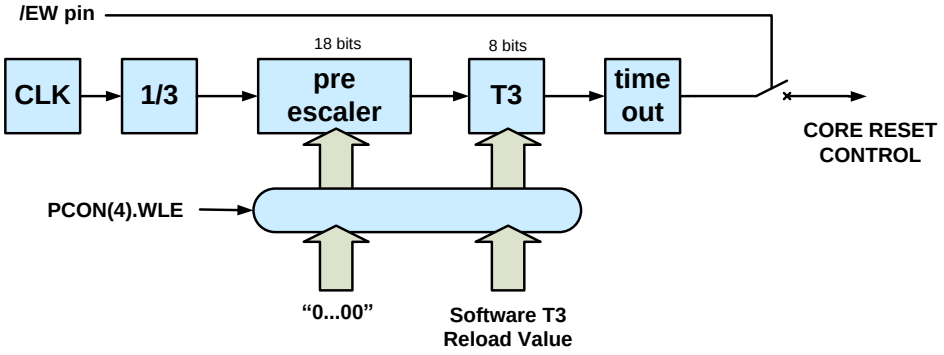
Reload of T3 is allowed only if WLE watchdog load enable (PCON(4)) is set.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
PCON	SMOD	--	--	WLE	GF1	GF0	PD	IDL

- SMOD: Doubles baud rate bit when timer 1 is used as time generator and serial port is in modes 1,2 or 3
- --: Reserved bit
- WLE: Watchdog load enable. It must be set by software to enable T3 reload. WLE is reset by hardware 13 machine cycles after been set by user software.
- GF1: General purpose flag bit, user programmable
- GF0: General purpose flag bit, user programmable
- PD: Sets power down mode
- IDL: Sets idle mode

Reload of T3 register automatically resets the preescaler and WLE

Figure 5-4. Watchdog timer



6. Standard Serial Interfaces

The serial ports are full duplex; they can transmit and receive simultaneously. The serial port reception and transmission registers are both accessed at Special Function Register SBUF(0,1,2). Writing to SBUF loads the transmit register and starts transmission, while reading SBUF accesses a physically separate receive register. The reception register is buffered and it can start reception of a second byte before a previously received byte has been read from the register. When a second byte is fully received the first one is lost.

The serial port control and status register is the Special Function Register SCON(0,1,2).

This register contains the mode selection bits, the 9th data bit for transmit and receive (TB8 and RB8), and the serial port interrupt bits (TI and RI).

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SCON	SM0	SM1	SM2	REN	TB8	RB8	TI	RI

- SM(0:1):
  - "00": Mode 0: Shift register mode; baud rate=1/12 fclk
  - "01": Mode 1: 8-bit UART mode; baud rate= T1 overflow
  - "10": Mode 2: 9-bit UART mode; baud rate= 1/32 or 1/64 fclk
  - "11": Mode 3: 9-bit UART mode; baud rate= T1 overflow
- SM2: Reception control. In modes 2 and 3 with SM2='1', RI will be '0' if the 9th received bit is '0'; In mode 1 with SM2='1', RI will be '0' if the received stop bit is not correct; In mode 0 SM2 keeps a '0'.
- REN: Serial port reception enable
- TB8: 9th transmitted bit in modes 2 and 3
- RB8: 9th received bit in modes 2 and 3, received stop bit in mode 1 when SM2='0'
- TI: Transmission interrupt flag. Set by hardware at the end of transmission, must be cleared by software
- RI: Reception interrupt flag. Set by hardware at the end of transmission, must be cleared by software

The serial port can operate in 4 modes. Transmission is always initiated by any instruction that writes SBUF, and TI flag is set after transmission. Reception is initiated in Mode 0 by the condition RI=0 and REN=1, and in the other modes by the incoming start bit if REN=1. In Modes 2 and 3, 9 data bits are received. The 9th one goes into RB8 and then comes a stop bit.

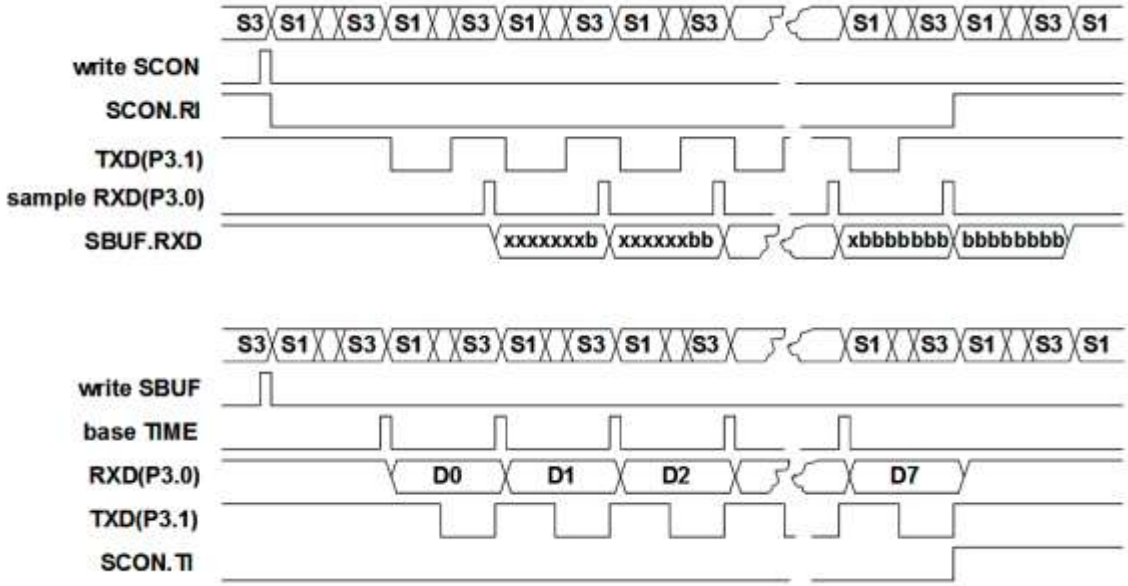
The port can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB8 = 1. This feature is enabled by setting bit SM2 in SCON.

6.1 Serial Port modes

6.1.1 Mode 0 (shift register mode)

Serial data inputs and exits through RxD. TxD outputs the shift clock. Eight bits are transmitted/received (LSB first). The baud rate is fixed at 1/12 the oscillator frequency. The internal timing is such that four full machine cycles will elapse between "write to SBUF" command and activating the transmission; transmission ends in the 36th machine cycle after "write to SBUF". Reception is initiated by the conditions REN = 1 and RI = 0, and after 36 machine cycles reception finish and RI is set to '1'.

Figure 6-1. Serial Port reception/emission mode 0



6.1.2 Mode 1 (8-bit UART)

In this mode, 10 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), and a stop bit (1). In reception, the stop bit goes into RB8 in SFR SCON. The baud rate is controlled by T1 rollover. Transmission is initiated by any instruction that uses SBUF as a destination register and the bit times are synchronized to T1 rollover. Reception is initiated by a detected 1-to-0 transition at RxD, RxD is sampled at a rate of 16 times each bit time and the value accepted is the value that was seen in at least 2 of the 3 central samples (7th, 8th, and 9th).

A correct reception loads SBUF and RB8 and sets RI, this is achieved when at the end of the receive process the condition (RI='0' and (SM2='0' or received_stop_bit='1')) is met. Otherwise, if the condition is not met, the received frame is irretrievably lost.

6.1.3 Modes 2 and 3 (9-bit UART)

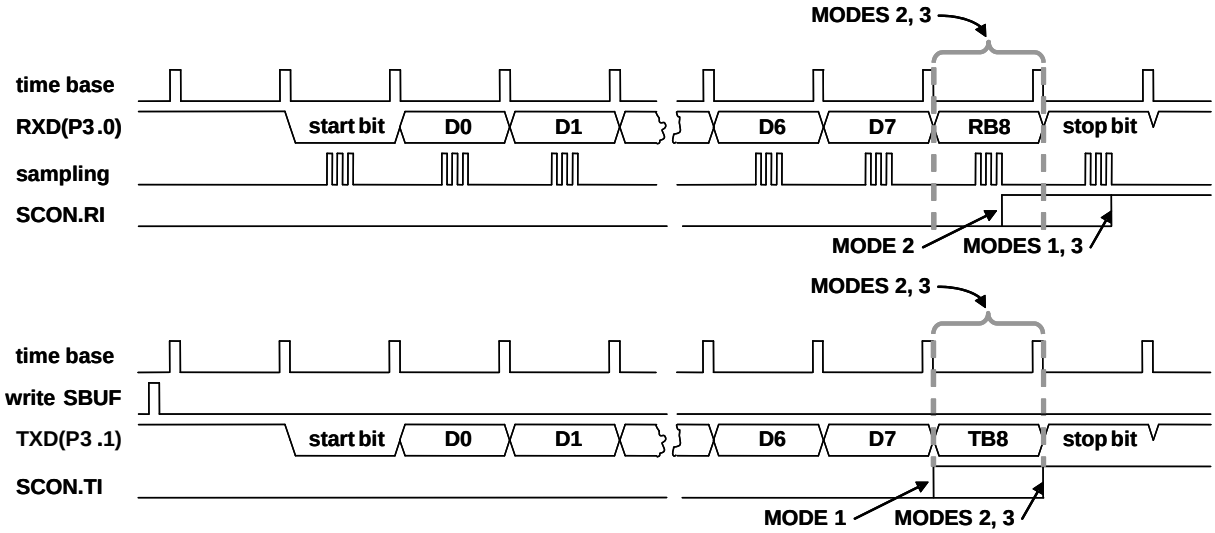
In this mode, 11 bits are transmitted (through TxD) or received (through RxD): start bit (0), 8 data bits (LSB first), a programmable 9th data bit (TB8), and a stop bit (1). The 9th transmitted data bit is loaded with TB8 value from SFR SCON. On reception, the 9th data bit goes into RB8 in SFR SCON. The stop bit is ignored. The baud rate is programmable to 1/32 or 1/64 the oscillator frequency. Mode 3 is the same as Mode 2 with programmable baud-rate controlled by T1 rollover.

The transmission begins with a "write to SUBF" instruction and finish at the 11th rollover after "write to SUBF", TI is set when transmission ends.

Reception is initiated by detecting a high to low level transition at RxD. For this purpose RxD is sampled at a rate of 16 times each bit. The value accepted is the value that was seen in at least 2 out of 3 central samples. When reception ends SBUF and RB8 are loaded and RI is set. A frame is correctly received if, and only if, the following condition is met at the time the last bit is received (RI=0 and (SM2='0' or 9^{th}_data_bit=1).

Otherwise, if the condition is not met the received frame is irretrievably lost and RI is not set.

Figure 6-2. Serial Port reception/emission modes 1,2,3.



6.2 Serial Port Timers (Timers 11, 12 and 14)

Timers 11, 12 and 14 are similar to T1 configured as 8-bit timer in auto reload mode (mode 2). These timers are used as baud rate generators for uarts 0, 1, and 2.

These timers are **not connected to the interruption system hardware**. They have not preescaler counter and their values are incremented every machine cycle (3 clock cycles).

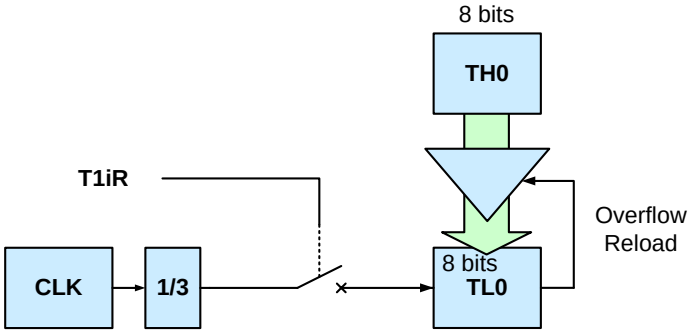
They can be used only as baud rate generators for the standard serial interfaces. When the run control bit in T1NC register (T1iR) is set the corresponding timer (T1i) starts counting machine cycles and the related SSI is controlled by the overflow of T1i.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
T1NC	WT12	WT11	WT10	--	T14R	--	T12R	T11R

- WT1(2:0): read/write access control to T1 timer registers
  - “000”: TH1, TL1
  - “001”: TH11, TL11
  - “010”: TH12, TL12
  - “100”: TH14, TL14
- --: Reserved bit.
- T14R: TT4 run control, when set T14 runs and T14 overflow controls UART2 variable baud rates.
- T12R: TT2 run control, when set T12 runs and T12 overflow controls UART1 variable baud rates.
- T11R: TT1 run control, when set T11 runs and T11 overflow controls UART0 variable baud rates.

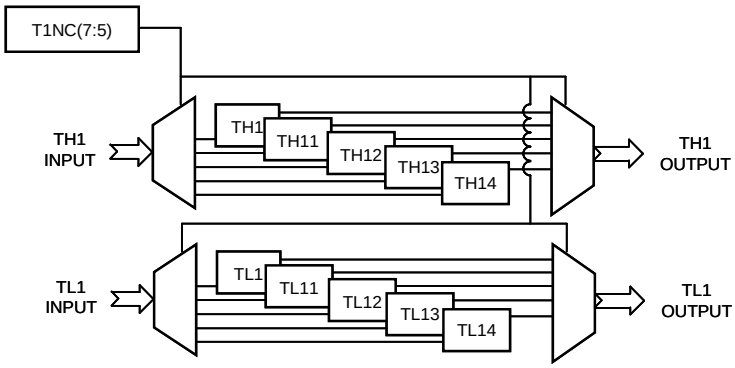
Registers TH11, TH12 and TH14 are mapped in the same address than TH1, to select the specific register to be accessed (read or write) the bits WT1(2:0) in T1NC register must be set to the desired value. TL11, TL12, TL14 and TL1 are accessed in the same way.

Figure 6-3. T1i timer



Timers T1i count machine cycles (3 clock cycles) in normal operation mode. They can be forced to count clock cycles. To enter this mode the corresponding bits in T1NP register must be set to ‘1’ (T1NP(i-1) controls T1i). Unused bits in T1NP, bits 4 to 7, are not defined. Writing to these bits has no effect, reading these bits will always get “0000”.

Figure 6-4. TH1 and TL1 registers



Mode 0

The baud rate in Mode 0 is fixed to Oscillator Frequency / 12.

Modes 1 and 3

The baud rates in Modes 1 and 3 are determined by the Timer 1 overflow rate and the value of SMOD and BAUD2 as follows:

Mode 2

The baud rate in Mode 2 depends on the value of bit SMOD in Special Function Register PCON and BAUD2 in Special Function Register CONF.

If [BAUD2,SMOD] = “00” (which is the value on reset), the baud rate is 1/64 the oscillator frequency.

If [BAUD2,SMOD] = “01” or “10”, the baud rate is 1/32 the oscillator frequency.

If [BAUD2,SMOD] = “11”, the baud rate is 1/16 the oscillator frequency

In the most typical applications, it is configured for “timer” operation in the auto-reload mode. In that case the baud rate is given by the formula:

Table 6-1 lists various commonly used baud rates and how they can be obtained using Timer1.

When timers T11, T12, T13 or T14 are selected as timing sources the baud rate is given by the formula:

Timers T1i count machine cycles (3 clock cycles) in normal operation mode. They can be forced to count clock cycles. To enter this mode the corresponding bits in T1NP register must be set to ‘1’. When T1i are configured in this mode the baud rate is given by the formula:

In this mode 0xFF and 0xFE are not allowed as reload values. When these values are used the behavior of USART is unpredictable.

Timer 2 overflow can also be used as baud rate generator for serial ports in modes 1 and 3. See timer 2 section for more information.

Selection of T1i as timing source overrides T2 and T1 sources. Selection of T2 as timing source overrides T1 source.

Table 6-1. **Timer 1 baud rates configuration examples**

Baud rate	Fosc	BAUD2	TPR(5:4)	SMOD	Timer 1		
					C/T	**MODE**	**Reload value**
153600	11059200	1	00	1	0	2	FDH
76800	11059200	0	00	1	0	2	FDH
38400	11059200	0	01	1	0	2	FDH
25600	11059200	0	10	1	0	2	FDH
19200	11059200	0	11	1	0	2	FDH
12800	11059200	0	10	0	0	2	FDH
9600	11059200	0	11	0	0	2	FDH
6400	11059200	0	10	0	0	2	FAH
4800	11059200	0	11	0	0	2	FAH
2400	11059200	0	11	0	0	2	F4H
1200	11059200	0	11	0	0	2	E8H

7. Serial Peripheral Interfaces SPI0 and SPI1

Two Serial Peripheral Interfaces SPI0 and SPI1 are implemented in ATPL00B.

Both SPI0 and SPI1 are full-duplex, synchronous communication bus with two operation modes: master mode and slave mode. When acting as an SPI master, the microcontroller supports baud rates of up to ¼ of the clock frequency.

The main difference between SPI0 and SPI1 is that while SPI0 is a byte oriented SPI, SPI1 is a byte-buffered SPI. SPI1 byte-buffering will be explained as a special SPI capability type in section 7.3. Ignoring this capability, both SPI0 and SPI1 are internally identical.

7.1 SPI description

During a byte transfer over SPI, the master and the slave simultaneously transmit (shift out) and receive (shift in) data. Master is the responsible in SPI system for transfer initialization, proper shifting and sampling of data. Every activity in the SPI environment is synchronized with the clock signal which is under master's control. A slave select line allows the master to select the desired slaves to exchange data with; not selected slave devices cannot interfere ongoing bus activities. In systems with multiple masters, slave select line on the master device can be used to detect multiple master bus contention.

SPI0 is configured and controlled using three special function registers: SPCTL, SPSTAT and SPDAT. (Similarly, SPI1 is configured and controlled by means of SPCTL1, SPSTAT1 and SPDAT1N registers)

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SPCTL	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	PSC1	PSC0

- SSIG: /SS0 slave select ignore, if set to '1' MSTR decides whether the device is a master or a slave and /SS0 pin can be used as port pin; if set to '0' and MSTR is set to '1' then the /SS pin decides whether the device is a master or a slave
- SPEN: SPI0 enabled '1' or disabled '0', when the SPI0 is disabled the SPI0 pins can be used as general I/O pins
- DORD: SPI0 data order

  '1' - the LSB of the data byte is transmitted first

  '0' - the MSB of the data byte is transmitted first
- MSTR: master '1' / slave '0' mode select. See SSIG field.
- CPOL: SPI0 clock polarity

  '1' – SPICLK0 is high when idle, the leading edge of SPICLK0 is the falling edge and the trailing edge is the rising edge

  '0' – SPICLK0 is low when idle, the leading edge of SPICLK0 is the rising edge and the trailing edge is the falling edge
- CPHA: SPI0 clock phase select

  '1' - data is driven on the leading edge of SPICLK0 and is sampled on the trailing edge

  '0' - the first data bit is on the MOSI0/MISO0 line before the first SPICLK0 leading edge, data is sampled on the leading edge of SPICLK0 and driven on the trailing edge of SPICLK

- PSC(1:0): SPI0 clock rate select, determines master clock output. 8 different baud rates can be selected using PSC2[SPSTAT(5)] & PSC(1:0)[SPCTL(1:0)].

  When working as slave, PSC(2:0) value is not taken into account, nevertheless this value must be different from “000”

PSC(2:0)	**clock rate**
“000”	fosc/4
“001”	fosc/16
“010”	fosc/64
“011”	fosc/128
“100”	fosc/256
“101”	fosc/512
“110”	fosc/1024
“111”	fosc/2048

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SPSTAT	SPIF	WCOL	PSC2	--	--	--	--	--

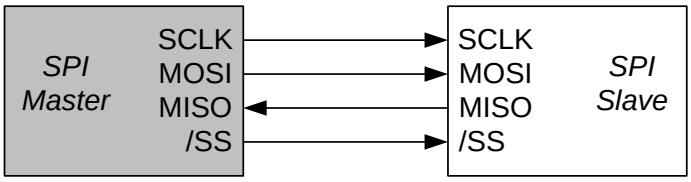
- SPIF: SPI0 transfer completion flag. When ESPI (IE.5) bit and EA (IE.7) bit are '1' and a SPI0 transfer finishes then the SPIF bit is set by hardware and an interrupt is generated. SPIF will also be set when SPI0 is in master mode with SSIG(SPCTL.7)='0' and /SS0(slave select) pin is driven low
- WCOL: SPI0 write collision flag. The WCOL bit is set by hardware when SPDAT is written during a data transfer. WCOL flag is cleared in software by writing '0' to this bit
- PSC2: SPI0 master clock rate select msb
- --: Reserved bit
-

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SPDAT	B7	B6	B5	B4	B3	B2	B1	B0

- B(7:0): Data transferred. B7 msb; B0 lsb.

The SPI interface requires four pins: SPICLK, MOSI, MISO and SS:

Figure 7-1. SPI connection diagram



SPI0 pins are mapped to P5 port pins as alternate output functions

Table 7-1. **SPI0 port map**

ADD8051C3A Port	Alternate Function
P5.0	/SS0
P5.1	SPICLK0
P5.2	MOSI0
P5.3	MISO0

SPICLK, MOSI and MISO are typically tied together between two or more SPI devices. Data flows from master to slave on the MOSI (Master Out Slave In) pin and flows from slave to master on the MISO (Master In Slave Out) pin. The SPICLK signal is output in the master mode and is input in the slave mode. If the SPI system is disabled, i.e. SPEN (SPCTL.6) = 0 (reset value), these pins can be used as general purpose I/O pins.

SS is an optional slave select pin. In a typical configuration, an SPI master asserts one of its port pins to select one SPI device as the current slave. An SPI slave device uses its SS pin to determine whether it is selected or not. The SS is ignored if any of the following conditions are true:

- The SPI0 system is disabled, i.e. SPEN (SPCTL.6) = 0 (reset value)
- SPI0 is enabled but SS pin is not needed in such system, i.e. SSIG(SPCTL.7) = 1; in this case SS pin can be used as general purpose pin on P5.

7.1.2 SPI clock phase, polarity and operation

There are four combinations (CPHA, CPOL) for sampling and shifting activities on data and clock lines in the SPI. Master can switch between any of them at any time thus having ability to communicate with slaves supporting data transfer using different modes.

In order to have successful data transfer between SPI devices, proper selection of the SPI clock phase and polarity is crucial. It is important to note that some of these configurations offer less capabilities than others. Clock Phase Bit CPHA allows user to specify the edges for sampling and shifting data. Clock Polarity bit CPOL allows user to set the clock polarity Figure 7-2 and Figure 7-3 show transfers with different values of CPHA and CPOL.

The SPI clock preescaler selection uses the PSC1-PSC0 bits in the SPCTL register and PSC2 in SPSTAT register. Master selects one of the available baud rates for the SPI communication. SPI Clock Preescaler bits do not have affect on the part acting as a slave, since it uses SPI clock supplied by the master.

When microcontroller operates as a slave with CPHA=’0’, some restrictions are present.

- SSIG must be ‘0’ and the SS pin must be negated and reasserted between each successive byte transfer.
- If the SPDAT register is written while SS is active (low), a write collision error results.
- Microcontroller’s behavior is undefined if CPHA is ‘0’ and SSIG is ‘1’.

On the other hand, slave having CPHA=’1’ may set SSIG to ‘1’. If SSIG = 1, the SS pin may remain active low between successive transfers (can be tied low at all times). This format is sometimes preferred in systems having a single fixed master and a single slave driving the MISO data line. Microcontroller configured as a master with CPHA=’0’ does not need to negate and reassert slave’s SS line in order to send and receive byte(s) of data.

In SPI, transfers are always initiated by the master. If the SPI is enabled (SPEN = 1) and microcontroller is configured as SPI master, writing to the SPI data register by the master will start the SPI clock generator and data transfer. The data will start to appear on MOSI about one half SPI bit-time to one SPI bit-time after data is written to SPDAT.

Figure 7-2. SPI0 transfer with CPHA=0

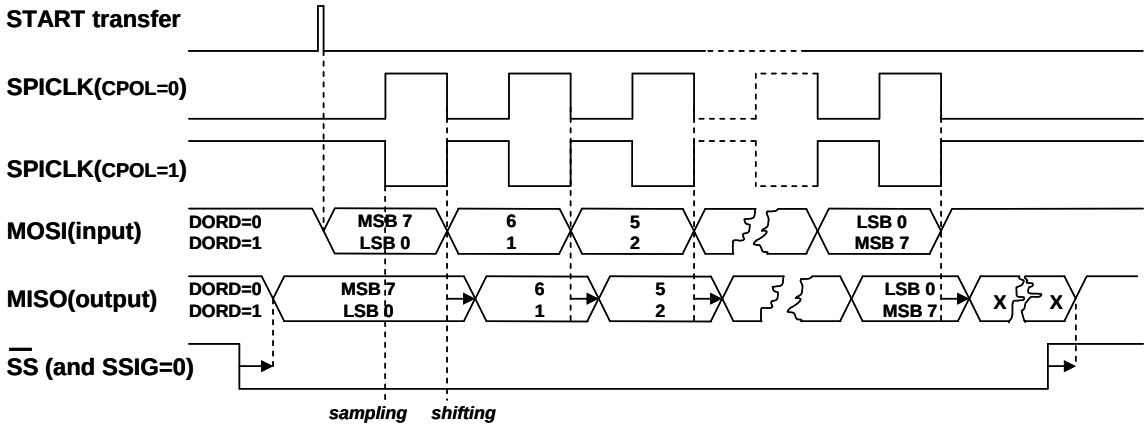
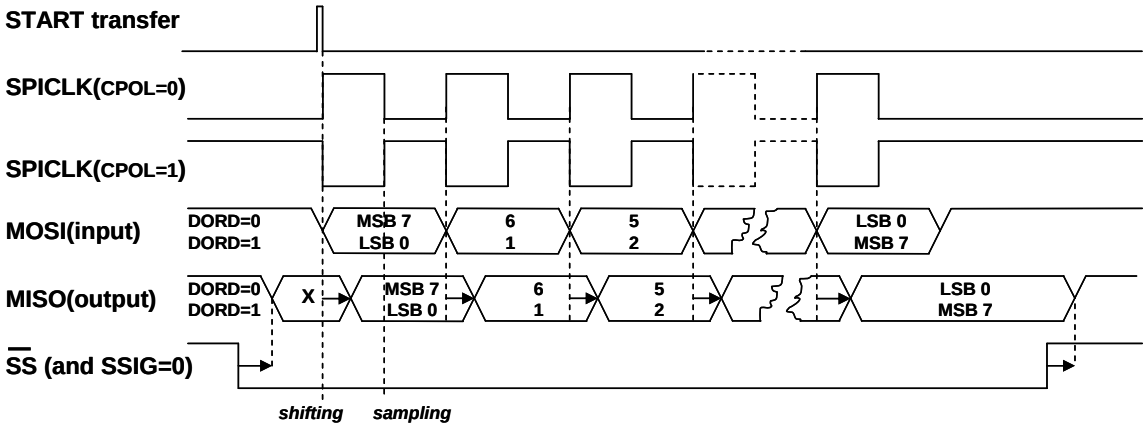


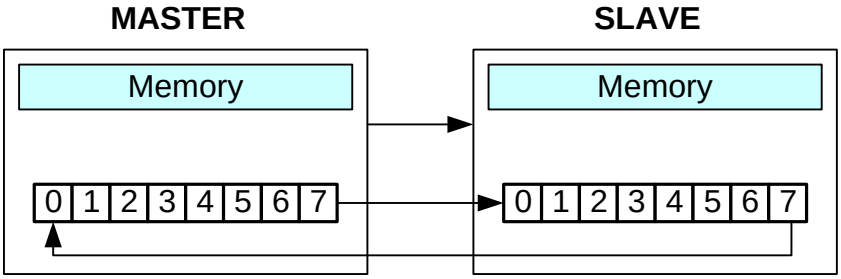
Figure 7-3. SPI0 transfer with CPHA=1



Note that the master selects a slave by driving the slave select pin of the corresponding slave device. Data written to the SPDAT register of the master is shifted out of the MOSI pin of the master to the MOSI pin of the slave, at the same time the data in SPDAT register on slave side is shifted out on its MISO pin to the MISO pin of the master.

After shifting one byte, the SPI clock generator stops, setting the transfer completion flag (SPIF) and an interrupt will be created if the SPI interrupt is enabled (ESPI, or IEN1.3 = 1). The two shift registers in the master CPU and slave CPU can be considered as one distributed 16-bit circular shift register. When data is shifted from the master to the slave, data is also shifted in the opposite direction simultaneously. This means that during one shift cycle, data in the master and the slave are interchanged.

Figure 7-4. SPI shift registers



If SPEN = 1, SSIG = 0 and MSTR = 1, the SPI0 is enabled in master mode. The SS pin is quasi-bidirectional. In this case, another master can drive this pin low to select this device as an SPI slave and start sending data to it. To avoid bus contention, the CPU becomes a slave. As a result of the observed SPI module becoming a slave, the MOSI and SPICLK pins are forced to be an input and MISO becomes an output.

The SPIF flag in SPSTAT is set, and if the SPI interrupt is enabled an SPI interrupt will occur.

User software should always check the MSTR bit. If this bit is cleared by a slave select and the user wants to continue to use the SPI as a master, the user must set the MSTR bit, otherwise it will stay in slave mode.

7.1.3 SPI0 write collision

The SPI0 is single buffered in the transmit direction and double buffered in the receive direction. New data for transmission cannot be written to the shift register until the previous transaction is completed. The WCOL (SPSTAT.6) bit is set to indicate data collision when the data register is written during transmission. In this case, the data currently being transmitted will continue to be transmitted, but the new data, i.e., the one causing the collision, will be lost.

While write collision is detected for either a master or a slave, it is uncommon for a master because the master has full control of the transfer in progress. The slave, however, has no control over transmission start by the master and therefore collision can occur.

Receiver transfers received data into a parallel read data buffer so that the shift register is free to accept a second character. However, the received character must be read from the Data Register before the next character has been completely shifted in. Otherwise. the previous data is lost. WCOL can be cleared in software by writing a ‘0’ to this bit.

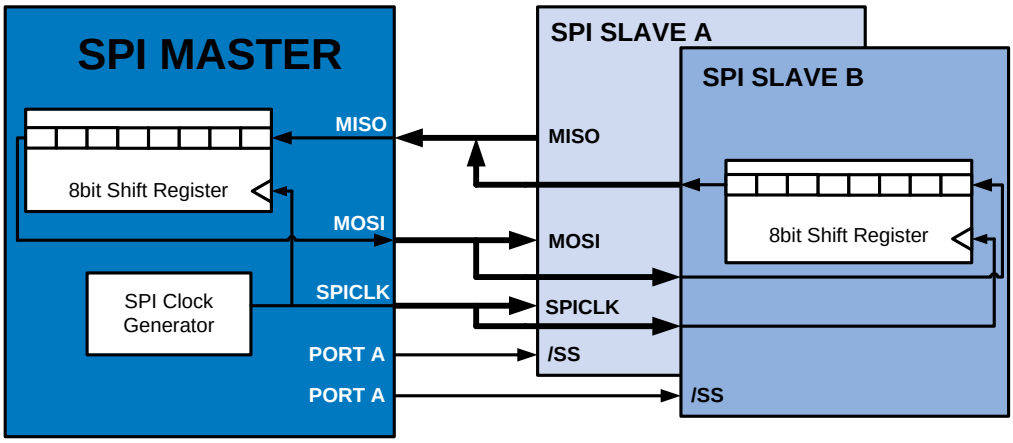
7.2 SPI Modes

Tha communication using SPI in a single master system is simple and usually works as described below:

- Both master and slave(s) configure their SPIs to operate in the same mode (one of four available modes) and turn them on; additionally, the master has to select baud rate for the communication; slave uses master’s serial clock to sample and shift data during the transfer
- Master selects desired slave unit(s) using its/their SS line(s)
- As soon as the master writes to its SPI buffer register (assuming that slave(s) has/have already loaded data into its/their buffer(s)), transfer is initiated; master’s SPI module generates serial clock and master’s and slave’s data are exchanged
- After the end of transfer, an indication is generated in all participating SPI units, both the master and the slave(s) read received data from their buffers
- If there is more data to be exchanged, all units taking part in this communication prepare the new set of data and master initiates another round of transfer; if there is no more data to be exchanged, master deselects used SS line(s) and previously active slaves are deselected; this is the end of SPI transfer

Typical connection in system using SPI is shown below.

Figure 7-5. SPI single master with multiple independent slaves configuration



In Figure 7-5, slave's SSIG = 0, and SS is used to select the slave. The SPI0 master can use any port pin (including P5.0/SS with SSIG = 1) to drive the SS pin.

Multimaster configuration is the case where two devices are connected to each other and either device can be a master or a slave. When no SPI0 operation is occurring, both can be configured as masters (MSTR = 1) with SSIG = 0 and P5.0 (SS) being in quasi-bidirectional mode. Before device initiates a transfer, it must set SSIG = 1 in order to avoid its own mode change since it will drive P5.0 low, forcing a mode change in the other device to slave.

Table 7-2 shows configuration for the master/slave modes in various cases.

Table 7-2. **SPI modes**

SPEN	SSIG	SS pin	MSTR	operation	MISO	MOSI	SPICLK	comment
0	X	P5.0(1)	X	disabled	P5.3(1)	P5.2(1)	P5.1(1)	SPI0 disabled, P5(3:0) are used as port pins
1	0	0	0	slave	out	in	in	selected as slave
1	0	1	0	slave	H	in	in	not selected, MISO is H to avoid bus contention
1	0	0	1(=>0) (2)	slave	out	in	in	the uC is configured as a master, then it is selected by SS to be a slave, MSTR bit is cleared and the uC becomes a slave
1	0	1	1	master	in	out	out	
1	1	P5.0(1)	0	slave	out	in	in	slave not needing SS
1	1	P5.0(1)	1	master	in	H	H	master not needing SS

Notes: 1. function as port or as alternate output function

2. the MSTR bit changes to '0' automatically when SS becomes low in input mode and SSIG is 0

7.3 SPI1 buffer operation

SPI1 is a byte buffered serial peripheral interface. Data transfers have a selectable additional number of bytes ranging from 0 to 7, SPSTAT1(2:0) is used to select the number of additional bytes. The buffer is composed of the registers SPDAT10 to SPDAT17. The content of SPDAT10 is the first to be transferred.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SPCTL1	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	PSC1	PSC0

- SSIG: /SS1 slave select ignore, if set to '1' MSTR decides whether the device is a master or a slave and /SS1 pin can be used as port pin; if set to '0' and MSTR is set to '1' then the /SS1 pin decides whether the device is a master or a slave
- SPEN: SPI1 enabled '1' or disabled '0', when the SPI1 is disabled the SPI1 pins can be used as general I/O pins
- DORD: SPI1 data order

  '1' - the LSB of the data byte is transmitted first

  '0' - the MSB of the data byte is transmitted first
- MSTR: master '1' / slave '0' mode select. See SSIG field
- CPOL: SPI1 clock polarity

  '1' – SPICLK1 is high when idle, the leading edge of SPICLK1 is the falling edge and the trailing edge is the rising edge

  '0' – SPICLK1 is low when idle, the leading edge of SPICLK1 is the rising edge and the trailing edge is the falling edge
- CPHA: SPI1 clock phase select

  '1' - data is driven on the leading edge of SPICLK1 and is sampled on the trailing edge

  '0' - the first data bit is on the MOSI1/MISO1 line before the first SPICLK1 leading edge, data is sampled on the leading edge of SPICLK1 and driven on the trailing edge of SPICLK1
- PSC(1:0): SPI1 clock rate select, determines master clock output. 8 different baud rates can be selected using PSC2[SPSTAT1(5)] & PSC(1:0)[SPCTL1(1:0)]

  When working as slave, PSC(2:0) value is not taken into account, nevertheless this value must be different from “000”

PSC(2:0)	**clock rate**
“000”	fosc/4
“001”	fosc/16
“010”	fosc/64
“011”	fosc/128
“100”	fosc/256
“101”	fosc/512
“110”	fosc/1024

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
SPSTAT1	SPIF	WCOL	PSC2	--	--	N2	N1	N0

- **--:** Reserved bit
- **SPIF:** SPI1 transfer completion flag. When ESPI1 (IE2.3) bit and EA (IE.7) bit are '1' and a SPI1 transfer finishes then the SPIF bit is set by hardware and an interrupt is generated. SPIF will also be set when SPI1 is in master mode with SSIG(SPCTL1.7)='0' and /SS(slave select) pin is driven low
- **WCOL:** SPI1 write collision flag. The WCOL bit is set by hardware when SPDAT is written during a data transfer. WCOL flag is cleared in software by writing '0' to this bit.
- **PSC2:** SPI1 master clock rate select msb
- **N(2:0):** Number of additional bytes to transfer in SPI1

The data bytes to be transferred must be written in order. Transfer will start when SPDAT1N is written if N=SPSTAT(2:0).

A write collision will occur if any SPDAT1N (with N<=SPSTAT(2:0)) is written before the 1+N bytes transfer is finished.

Table 7-3. **SPI1 port map**

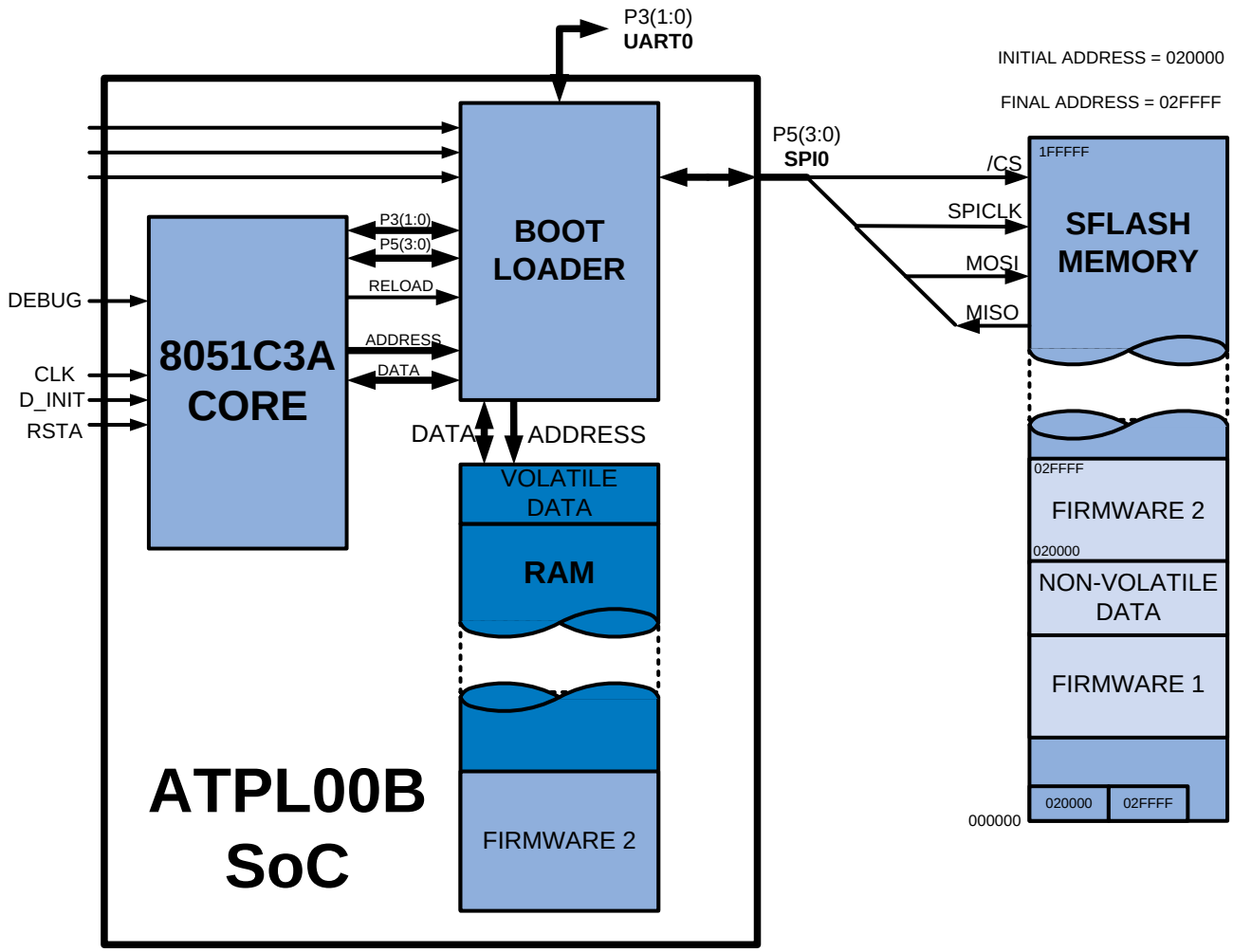
ADD8051C3A Port	Alternate Function
P4.2	/SS1
P4.3	SPICLK1
P4.4	MOSI1
P4.5	MISO1

8. Boot Loader

ATPL00B needs an external SPI flash memory to store the firmware and all no-volatile data, while an internal 128Kbytes SRAM is used for firmware execution (once a volatile firmware copy has been automatically generated by the boot loader in the system startup process) and to store all the volatile variables.

After power-up the boot loader forces a start-up cycle uploading the target program from the serial flash to the internal SRAM, the program is executed from the lower entries of the internal SRAM and the upper entries are used to store program data. A boot loader diagram is shown in Figure 8-1

Figure 8-1. Boot-loader diagram



Boot loader is executed in the following situations:

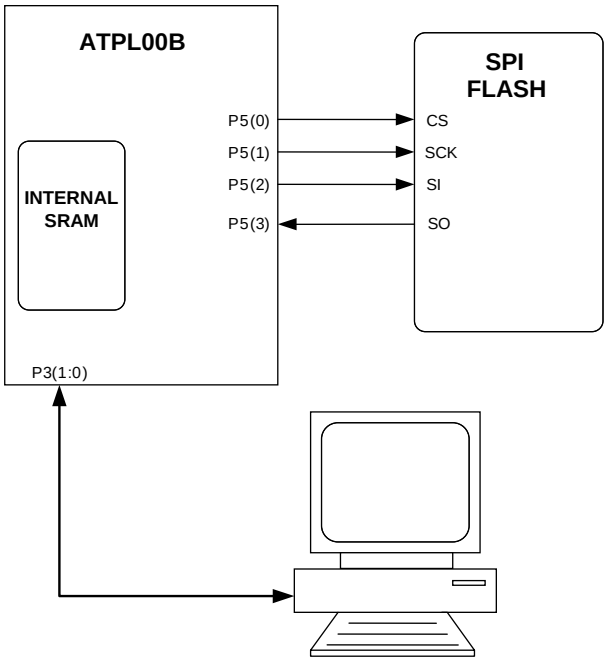
- Every time after a Power-On.
- Every time the system is reset by the asynchronous reset signal (RSTA).
- Every time the bit 2 in CONF SFR is set, forcing a reload.
- After a Watchdog timeout, only if bit 1 in CONF SFR is set.

The boot loader circuitry also supports in-system programming. A user can update the firmware via a serial interface without additional logic and without disassembling the system.

When a SPI flash with serial number or/and a Silicon Serial Number IC are available in the system board, the system supports automatic firmware encryption in the flash.

In the ATPL00B the microcontroller can force program reloading by writing in CONF Specific Function Register.

Figure 8-2. Flash connection diagram



8.1 Pin Description

- **/PROG:** After power up, if this signal is tied low the in-system programming mode is enabled, else the system is in execution mode.
  - ‘0’: In-system programming mode
  - ‘1’: Execution mode
- **SECURED:** This pin enables encrypted firmware execution when the board configuration supports it (see Encrypted firmware requirements section).
  - ‘0’: Cryptographic Storage disabled
  - ‘1’: Cryptographic Storage enabled
- **P1.7 (SSN):** Input pin used to read a Serial Number if a valid Silicon Serial Number device is being used (see 0)

As shown in Figure 8-1 the boot loader acts as a bus switcher. It takes control over the Standard Serial port 0 (P3[1:0]) and pins of port P5 (P5[3:0]) that are used as Serial Peripheral Interface to connect the serial flash (SPI0). See table below:

Table 8-1. **SS0 & SPI0 buses**

ADD8051C3A Port	Alternate Function
P3.0	RxD0
P3.1	TxD0
P5.0	/SS0
P5.1	SPICLK0
P5.1	MOSI0
P5.3	MISO0

After program booting, the shared pins are fully software controlled. This allows the microcontroller to store no-volatile program data to the serial flash. The microcontroller has full access to all the serial flash content which **has to be handled carefully to avoid unexpected overwriting of program code located in the flash memory.**

8.2 Flash Programming

8.2.1 In-System programming

The boot loader supports in-system programming for SPI flash memories.

To enable in-system programming /PROG pin must be tied low.

When in-system programming is enabled, the boot loader is controlled via RS-232 commands.

The default RS-232 configuration is 57600 bauds, 8 data bits, 1 stop bit and no parity.

Atmel provides a set of tools that allows customers to carry on in-system programming. For further information, please contact Atmel support.

8.2.2 SPI Flash programming

After a RESET, if /PROG pin is tied low, SPI flash commands are allowed. This set of commands allows the user to send any standard SPI command to the flash memory through the serial port. The boot loader acts as a bridge from serial to SPI protocol.

The serial commands are composed by a header for the boot loader and the SPI command itself.

Figure 8-3. Serial commands

- **ICODE:** This byte indicates to the boot loader the type of instruction to be executed

ICODE (Hex)	RS-232 FORMAT (Hex)
01	01 N-1 OPC AD2 AD1 AD0 D1 DN
02	02 N-1 OPC AD2 AD1 AD0
04	04 XX OPC
08	08 XX OPC AD2 AD1 AD0
10	10 N-1 OPC
20	20 N-1 OPC D1 DN

- **BC:** Byte Count indicates the number of data bytes to read or write. The value of this field must be the number of data bytes to read or write minus one, that is, the range of the BC field is [0,255] and the number of data bytes range is [1,256]. If the requested instruction is not a reading or writing instruction the BC field is not used and can take any value.

  Note: If Atmel SPI flash memory (see 8.3.2) is being used, and it's configured with a page size of 264 bytes, BC field will have a length of 2 bytes.

- **ADDR(2:0):** 24 bits address field. This field must point to the initial address of the instruction operation.

  DATA(1:n): Data bytes of the SPI instruction (n = BC + 1). The DATA1 byte is the first data byte in a write or read operation.

With the six command types listed in ICODE description, it is possible to execute any standard operation on the SPI flash memory:

Table 8-2. **ICODE operations**

ICODE (Hex)	Typical Use
01	Page programming
02	Read, Read id.
04	Write enable/disable
08	Erase
10	Read status register
20	Write status register

Example: Programming 256 bytes to flash memory starting at 010000 (Hex) address. (The page programming opcode of the memory is 02 (Hex) and the required erase procedure is not included in this example).

The RS-232 command in hexadecimal format is:

01.FF.02.01.00.00.[byte1 to byte256]

8.3 System startup

When the system is configured in execution mode (/PROG pin is set) and after a power up or reload process, the boot loader checks the system configuration and performs the required operations to ensure the correct execution of the firmware.

Boot loader transfers a volatile copy of the firmware from the flash memory to the SRAM memory starting at address 000000 (Hex). The internal SRAM size is 128Kbytes and the maximum SPI flash size supported is 16Mbits. The part of the SRAM that is not used to store the firmware is used by the system to store the volatile data in the execution process.

The volatile data is stored in the SRAM memory from top to bottom. This arrangement is done automatically by the system, that is, when the microcontroller wants to save data at address A the system automatically converts the target address to (top address – A). Using this storage method, the amount of volatile data that can be stored in the RAM is determined by the firmware size.

The transfer process can last about 400ms depending on flash device and code size. The first time this process will be longer if auto encryption is activated. Auto encryption must be done once, and **care must be taken about respecting encryption times**.

When the transfer process is finished, the microcontroller begins to execute the program and the system is ready to use. The startup cycle is performed each time the system is powered up or when the microcontroller via software or watchdog forces a firmware reload process writing in a specific register.

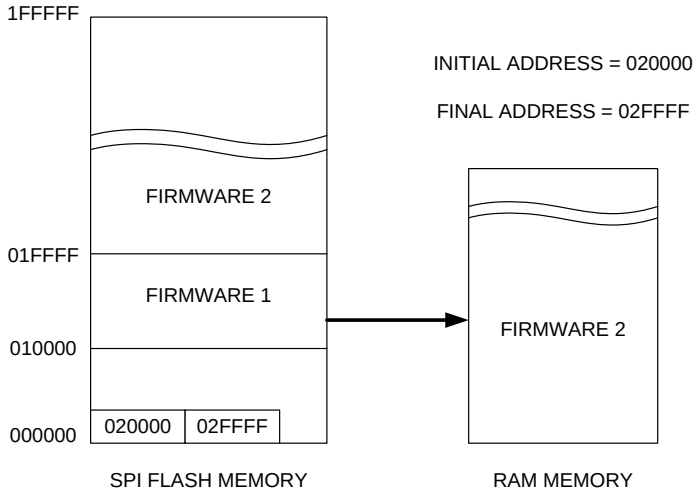
Both the location of the firmware at the SPI flash and the transfer size are configurable. The lower six bytes of the SPI flash are used to store the initial and final position of the firmware (from address 0x000000 to 0x000005).

This feature allows working with a SPI flash device that has multiple firmware versions stored in different addresses. The version to be used can be selected by means of its initial and final addresses.

The transfer size is equal to the firmware size.

Bank-switching is supported, so firmware code bigger than 64KB can be managed. Extreme caution must be taken when using bank-switching in order to respect code and variables spaces and avoid overlapping.

Figure 8-4. Flash & SRAM memories diagram



8.3.1 Encrypted firmware requirements

In order to run Auto-encryption procedure, the conditions below must be met:

- SECURED = ‘1’
- DEBUG = ‘0’
- /PROG = ‘1’
- Valid Serial Number (see supported devices section)
- Correct flash memory START and END addresses
- Detection by hardware of no-encrypted software stored in a valid flash device
- Starting address of the stored firmware multiple of 4KB (0xXXX000)

First time auto-encryption procedure runs, the code is encrypted and rounded to the next multiple of 4kbytes, and the executions begins.

Next time the system boots, hardware will detect encrypted software in the flash, thus auto encryption procedure will not be started (that is, auto encryption is only necessary to be performed once).

Note1: Auto encryption procedure takes about 30 seconds to be completed, and during encryption no external signals or LEDs are asserted, so care must be taken by the user and encryption time must be respected.

Note2: If SECURED = ‘0’ or some of the other conditions above are not met, the system may run without code encryption. Care must be taken by the user.

Note3: If SECURED = ‘0’ and the code stored in the flash is encrypted (an unusual situation), the system will be expecting no-encrypted code and will fail.

Note 4: A helpful way to check that the firmware in the flash has been encrypted, is to read the start and end addresses allocated at the beginning of the code. If encryption has been done, these address values will make no sense because of the encryption.

8.3.2 Supported Devices

The boot loader supports any SPI flash memory compatible with SPI mode 0 (CLK signal is normally low) and with a value of read instruction operation code equals to 0x03 (Hex). The maximum supported flash size is 16Mbits

Atmel AT45DBxx SPI Flash family is strongly recommended and has been fully tested in ATPL00B development kits and in field configurations.

8.3.2.1 Serial Number Device

If the system is configured to use encrypted firmware it is necessary to include in the system board a device that provides a serial number to the ATPL00B SoC.

Atmel Flash AT45DBxx device also includes its own Serial Number, so the flash device provides the Serial Number itself and there is not necessary an external SN chip.

When a more robust encryption is desired, a SN device can be added to the system, for example:

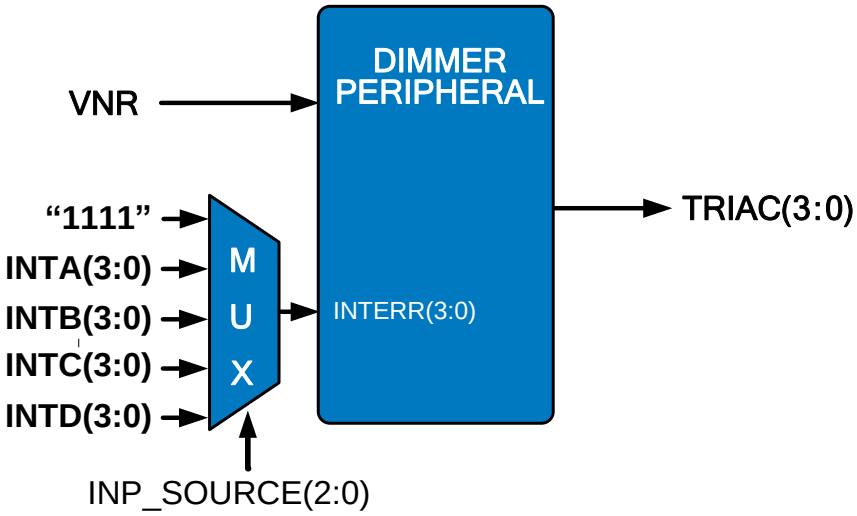
- DS2401 (Dallas): IC Silicon Serial Number. Provides a valid serial number to be used for encryption purposes.

When both devices are used together (AT45DBxx+DS2401), a more robust encryption is achieved.

9. Dimmer Peripheral

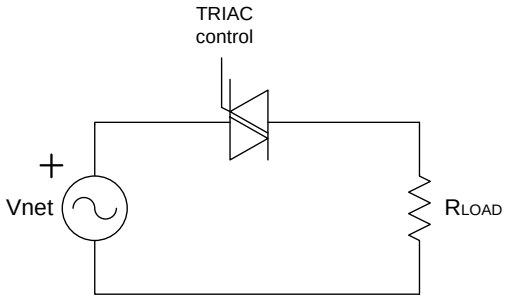
ATPL00B SoC contains a dimmer peripheral to control up to four triacs and four switches.

Figure 9-1. Dimmer Peripheral diagram



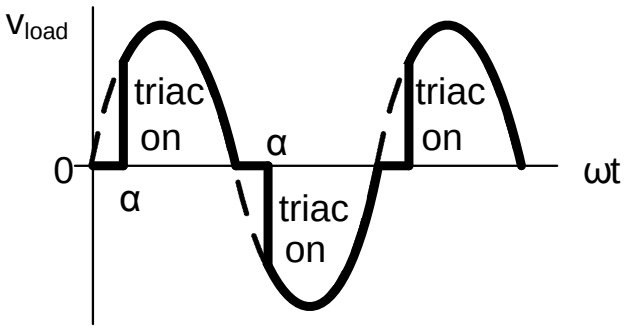
This peripheral is capable of doing a Phase Angle control in order to change the power of the loads. Phase Angle control uses a low switch frequency to chop the power line sine wave.

Figure 9-2. TRIAC control detail



The firing angle (α) of the switch can be varied. The average voltage will be proportional to the area under the sine wave. Thus, the average voltage is the integral from the firing angle to the zero crossing.

Figure 9-3. Firing angle



The configuration registers are used to change the firing angle, turn on/off the loads, and read the state of the switches.

This peripheral also can be used to generate a PWM (Pulse-Width Modulation) control. The adjustment of the period of PWM control signals is explained in more detail in 9.1.8

This control software tasks spend a high percent of CPU time, thus the ATPL00B performs some of these tasks via hardware in order to reduce the CPU computational load of the 8051 integrated microcontroller.

9.1 Configuration Registers

The configuration registers of the dimmer peripheral are accessed by the microcontroller as peripheral registers. Table 9-1 shows the configuration registers address map.

Table 9-1. **Dimmer Peripheral configuration registers**

Address	Reg. Name	Reset Value (hex)
0xFEA0	DIM_CTRL	0x06
0xFEA1	INP_ST	0x00
0xFEA2	OUT_ST	0x00
0xFEA3	OUT_REF3	0x00
0xFEA4	OUT_REF2	0x00
0xFEA5	OUT_REF1	0x00
0xFEA6	OUT_REF0	0x00
0xFEA7	POLARITY	0xFF
0xFEA8	D_CONF	0xFF
0xFEA9	V_SWC	0xFF
0xFEAA	PWM_PER	0x00
0xFEAB	INP_SOURCE	0x00

The default values after performing a reset on the dimmer peripheral configuration registers are also shown in the table above. Using the default values results in the following dimmer peripheral behavior:

- All loads are switched off.
- To calculate the zero-crossing only the VNR rising edge is used, and the middle point is calculated automatically
- Power frequency is 50Hz
- All triacs are fired with chip ground.
- All switches are detected non-inverted.
- All switches are high voltage type (AC switch)
- All switches are detected with an optocoupled connection (see Figure 9-8)
- PWM is not used
- All triacs are firing without a pulse train
- Switches are not used

9.1.1 DIM_CTRL register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
DIM_CTRL	--	--	--	--	FQ	VEZC	REZC	FECZ

Name: DIM_CTRL

Address: 0xFEA0

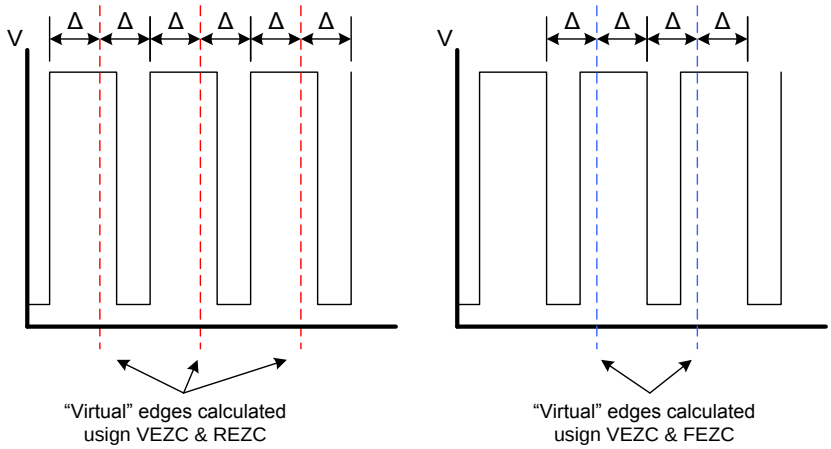
Reset: “00000110”

This register selects how to calculate the zero-crossing point depending on the external circuit type connected to VNR pin.

- **--:** Reserved bit
- **FQ:** Selects the mains frequency.
  - ‘0’: 50Hz frequency
  - ‘1’: 60Hz frequency
- **VECZ:** Virtual Edge for Zero Crossing

  In this bit is equal to one, the hardware calculates the middle point between two VNR edges to calculate de zero crossing.

  This mode is useful when the VNR signal duty cycle is different from 50%:



  VECZ can be used simultaneously with REZC or FECZ.

  Using the three of them at a time is not recommended.

- **REZC**: Rising Edge for zero crossing

  If this bit is set to ‘1’, the hardware uses the VNR rising edges to calculate zero-crossing.

  FEZC and REZC can be used simultaneously

- **FECZ**: Falling Edge for Zero Crossing-

  If this bit is set to ‘1’, the hardware uses the VNR falling edges to calculate zero-crossing.

  FEZC and REZC can be used simultaneously

9.1.2 INP_ST register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
INP_ST	--	--	--	--	D3	D2	D1	D0

Name: INP_ST

Address: 0xFEA1

Reset: “00000000”

This register contains the state of the switches.

This register is read-only.

- **D3:** Status of the switch connected to INTERR3 input
  - ‘0’: The switch connected to INTERR3 input is OFF
  - ‘1’: The switch connected to INTERR3 input is ON

  Note: The polarity of this field is set by IN_POL3 field in POLARITY register
- **D2:** Status of the switch connected to INTERR2 input
  - ‘0’: The switch connected to INTERR2 input is OFF
  - ‘1’: The switch connected to INTERR2 input is ON

  Note: The polarity of this field is set by IN_POL2 field in POLARITY register
- **D1:** Status of the switch connected to INTERR1 input
  - ‘0’: The switch connected to INTERR1 input is OFF
  - ‘1’: The switch connected to INTERR1 input is ON

  Note: The polarity of this field is set by IN_POL1 field in POLARITY register
- **D0:** Status of the switch connected to INTERR0 input
  - ‘0’: The switch connected to INTERR0 input is OFF
  - ‘1’: The switch connected to INTERR0 input is ON

  Note: The polarity of this field is set by IN_POL0 field in POLARITY register

9.1.3 OUT_ST register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
OUT_ST	--	--	--	--	D3	D2	D1	D0

Name: OUT_ST

Address: 0XFEA2

Reset: “00000000”

This register selects which loads are switched on. If PWM mode is selected, then this register is used to indicate which PWM outputs are active.

- **D3:** This bit sets the state of the load connected to TRIAC3 output
  - ‘0’: The load connected to TRIAC3 output is OFF
  - ‘1’: The load connected to TRIAC3 output is ON
- **D2:** This bit sets the state of the load connected to TRIAC2 output
  - ‘0’: The load connected to TRIAC2 output is OFF
  - ‘1’: The load connected to TRIAC2 output is ON
- **D1:** This bit sets the state of the load connected to TRIAC1 output
  - ‘0’: The load connected to TRIAC1 output is OFF
  - ‘1’: The load connected to TRIAC1 output is ON
- **D0:** This bit sets the state of the load connected to TRIAC0 output
  - ‘0’: The load connected to TRIAC0 output is OFF
  - ‘1’: The load connected to TRIAC0 output is ON

9.1.4 OUT_REF registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
OUT_REFi	ORi_D7	ORi_D6	ORi_D5	ORi_D4	ORi_D3	ORi_D2	ORi_D1	ORi_D0

Name: OUT_REF3 – OUT_REF0

Address: 0xFEA3 – 0xFEA6

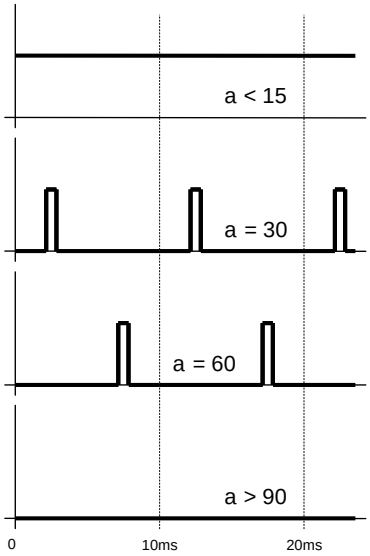
Reset: “00000000”, “00000000”, “00000000, “00000000”

These four registers are used to adjust the power of the loads connected to the TRIACS (if PWM mode is not selected) or the output duty cycle (when PWM mode is selected).

- **OR3_D(7:0):**This register is used to adjust the power of the load connected to TRIAC3 when PWM mode is not selected. When PWM mode is selected, this register controls the duty cycle.
- **OR2_D(7:0):**This register is used to adjust the power of the load connected to TRIAC2 when PWM mode is not selected. When PWM mode is selected, this register controls the duty cycle.
- **OR1_D(7:0):**This register is used to adjust the power of the load connected to TRIAC1 when PWM mode is not selected. When PWM mode is selected, this register controls the duty cycle.
- **OR0_D(7:0):**This register is used to adjust the power of the load connected to TRIAC0 when PWM mode is not selected. When PWM mode is selected, this register controls the duty cycle.

Description:

In case that PWM mode is not selected, the registers contain a value between 0 and 99 (0x00 to 0x63). If value is lower than 15 (0x0F), the firing is continuous. Otherwise, the firing width is 200 microseconds. With values greater than 90 (0x5A), the TRIAC is not fired. If values greater than 99 (0x63) are used, the TRIAC may be fired at the next semi period.



In case that PWM mode is selected, these four registers are used to adjust the duty cycle. In this case, each one must contain a value between 0 and 255 (0xFF). When 255 (0xFF), PWM output is always VDD.

9.1.5 POLARITY register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
POLARITY	IN_POL3	IN_POL2	IN_POL1	IN_POL0	OUT_POL3	OUT_POL2	OUT_POL1	OUT_POL0

Name: POLARITY

Address: 0xFEA7

Reset: “11111111”

- **IN_POL3:** This bit sets the polarity of D3 field in INP_ST register
  - ‘0’: The bit D3 will be equal to ‘1’ when the switch connected to INTERR3 input is OFF, and equal to ‘0’ when the switch is ON
  - ‘1’: The bit D3 will be equal to ‘0’ when the switch connected to INTERR3 input is OFF, and equal to ‘1’ when the switch is ON
- **IN_POL2:** This bit sets the polarity of D2 field in INP_ST register
  - ‘0’: The bit D2 will be equal to ‘1’ when the switch connected to INTERR2 input is OFF, and equal to ‘0’ when the switch is ON
  - ‘1’: The bit D2 will be equal to ‘0’ when the switch connected to INTERR2 input is OFF, and equal to ‘1’ when the switch is ON
- **IN_POL1:** This bit sets the polarity of D1 field in INP_ST register
  - ‘0’: The bit D1 will be equal to ‘1’ when the switch connected to INTERR1 input is OFF, and equal to ‘0’ when the switch is ON
  - ‘1’: The bit D1 will be equal to ‘0’ when the switch connected to INTERR1 input is OFF, and equal to ‘1’ when the switch is ON
- **IN_POL0:** This bit sets the polarity of D0 field in INP_ST register
  - ‘0’: The bit D0 will be equal to ‘1’ when the switch connected to INTERR0 input is OFF, and equal to ‘0’ when the switch is ON
  - ‘1’: The bit D0 will be equal to ‘0’ when the switch connected to INTERR0 input is OFF, and equal to ‘1’ when the switch is ON
- **OUT_POL3:** This bit sets if the triac connected to TRIAC3 output is firing whether with “chip ground” or VDD
  - ‘0’: The triac is firing with VDD
  - ‘1’: The triac is firing with chip ground
- **OUT_POL2:** This bit sets if the triac connected to TRIAC2 output is firing whether with “chip ground” or VDD
  - ‘0’: The triac is firing with VDD
  - ‘1’: The triac is firing with chip ground
- **OUT_POL1:** This bit sets if the triac connected to TRIAC1 output is firing whether with “chip ground” or VDD
  - ‘0’: The triac is firing with VDD
  - ‘1’: The triac is firing with chip ground
- **OUT_POL0:** This bit sets if the triac connected to TRIAC0 output is firing whether with “chip ground” or VDD
  - ‘0’: The triac is firing with VDD
  - ‘1’: The triac is firing with chip ground

9.1.6 D_CONF register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
D_CONF	OPTO3	OPTO2	OPTO1	OPTO0	SPF3	SPF2	SPF1	SPF0

Name: D_CONF

Address: 0xFEA8

Reset: “11111111”

This register selects the switches detector type and the triacs firing type

- **OPTO3:** This bit sets if the switch connected to INTERR3 input is being detected by means of an optocoupler or a Zener diode
  - ‘0’: Zener diode detection
  - ‘1’: Optocoupler detection
- **OPTO2:** This bit sets if the switch connected to INTERR2 input is being detected by means of an optocoupler or a Zener diode
  - ‘0’: Zener diode detection
  - ‘1’: Optocoupler detection
- **OPTO1:** This bit sets if the switch connected to INTERR1 input is being detected by means of an optocoupler or a Zener diode
  - ‘0’: Zener diode detection
  - ‘1’: Optocoupler detection
- **OPTO0:** This bit sets if the switch connected to INTERR0 input is being detected by means of an optocoupler or a Zener diode
  - ‘0’: Zener diode detection
  - ‘1’: Optocoupler detection
- **SPF3:** This bit sets if the load connected to TRIAC3 output is firing whether with a simple pulse or a pulse train.
  - ‘0’: Pulse train firing
  - ’1’: Simple pulse firing
- **SPF2:** This bit sets if the load connected to TRIAC2 output is firing whether with a simple pulse or a pulse train.
  - ‘0’: Pulse train firing
  - ’1’: Simple pulse firing
- **SPF1:** This bit sets if the load connected to TRIAC1 output is firing whether with a simple pulse or a pulse train.
  - ‘0’: Pulse train firing
  - ’1’: Simple pulse firing
- **SPF0:** This bit sets if the load connected to TRIAC0 output is firing whether with a simple pulse or a pulse train.
  - ‘0’: Pulse train firing
  - ’1’: Simple pulse firing

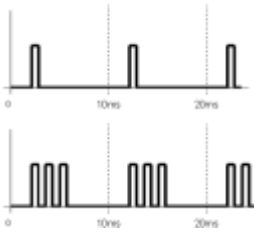


Figure 9-4. Simple pulse vs. Pulse train

9.1.7 V_SWC register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
V_SWC	HVI3	HVI2	HVI1	HVI0	PWM3	PWM2	PWM1	PWM0

Name: V_SWC

Address: 0XFEA9

Reset: “11111111”

This register selects the voltage switches type (AC alternating-current high voltage switch or DC direct-current low voltage switch) and activates the PWM (Pulse Width Modulation) control.

- **HVI3:** Selects the switch type (AC switch or DC switch) that is connected to INTERR3 (internal logic acts in different way depending on the switch type)
  - ‘0’: DC switch is connected
  - ’1’: AC switch is connected
- **HVI2:** Selects the switch type (AC switch or DC switch) that is connected to INTERR2 (internal logic acts in different way depending on the switch type)
  - ‘0’: DC switch is connected
  - ’1’: AC switch is connected
- **HVI1:** Selects the switch type (AC switch or DC switch) that is connected to INTERR1 (internal logic acts in different way depending on the switch type)
  - ‘0’: DC switch is connected
  - ’1’: AC switch is connected
- **HVI0:** Selects the switch type (AC switch or DC switch) that is connected to INTERR0 (internal logic acts in different way depending on the switch type)
  - ‘0’: DC switch is connected
  - ’1’: AC switch is connected
- **PWM3:** Sets the behavior of TRIAC3 output mode
  - ‘0’: PWM output
  - ’1’: Phase angle control output
- **PWM2:** Sets the behavior of TRIAC2 output mode
  - ‘0’: PWM output
  - ’1’: Phase angle control output
- **PWM1:** Sets the behavior of TRIAC1 output mode
  - ‘0’: PWM output
  - ’1’: Phase angle control output
- **PWM0:** Sets the behavior of TRIAC0 output mode
  - ‘0’: PWM output
  - ’1’: Phase angle control output

9.1.8 PWM_PER register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
PWM_PER	D7	D6	D5	D4	D3	D2	D1	D0

Name: PWM_PER

Address: 0xFEAA

Reset: “00000000”

This register is used to adjust the period of the PWM control

- **D(7:0):** Values can be between 0 and 255 (0xFF).
  The smaller the value, the greater the frequency

PWM FREQUENCY = ———— KHz

Where n=D(7:0) value

9.1.9 INP_SOURCE register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
INP_SOURCE	--	--	--	--	--	D2	D1	D0

Name: INP_SOURCE

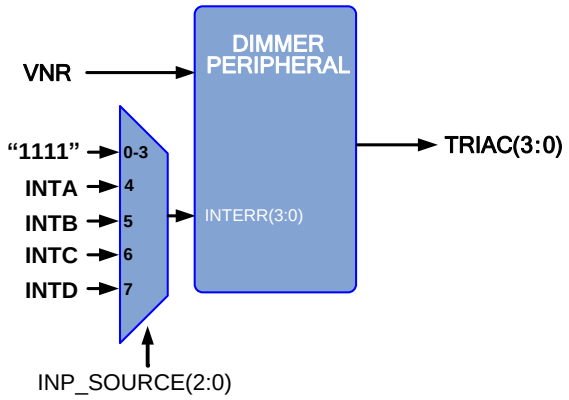
Address: 0xFEAB

Reset: “00000000”

- **--:** Reserved bit
- **D(2:0):** This register controls the multiplexor that selects the input to INTERR(3:0) (dimmer switches). See Figure 9-5

D(2:0)	INTERR(3:0) input	Comment
“000” (default)	“1111”	INTC(3:0) in “Hi-Z” INTC(3:0) in “Hi-Z”
“001”	“1111”	
“010”		
“011”		
“100”	INTA(3:0)	Microcontroller **must not** write a '0' in these ports when they are being used as switch inputs
“101”	INTB(3:0)	
“110”	INTC(3:0)	
“111”	INTD(3:0)	

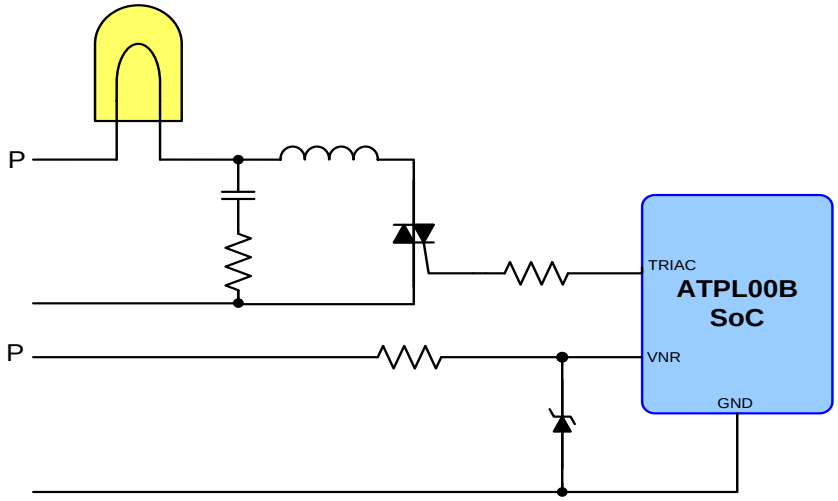
Figure 9-5. INTERR input mux



9.2 External Circuits

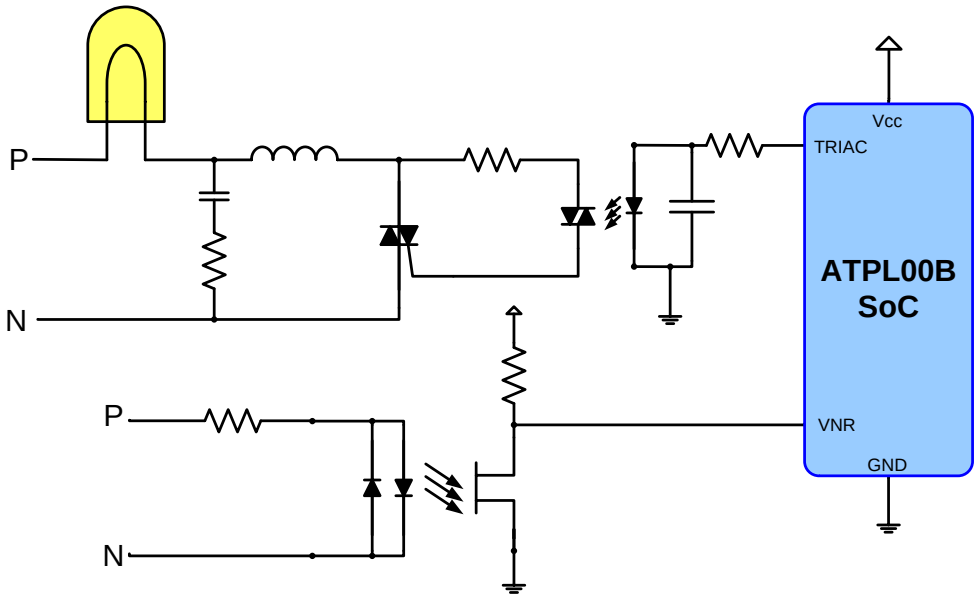
Some external components are necessary to control the loads connected to the ATPL00B. The external circuitry must be different depending if the integrated circuit GND pin is connected to power line Neutral or not. Examples of external circuit configurations are described below:

Figure 9-6. Phase Control with "non-isolated" connection



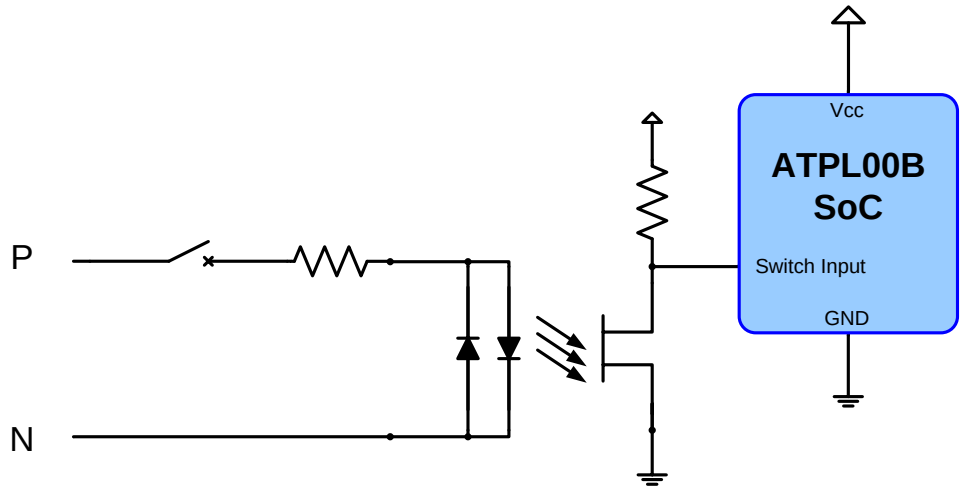
VNR pin is used to detect the power line wave zero crossing. If the chip ground is not connected to the power line (N) an optocoupler must be used in the detection circuit as shown in Figure 9-7.

Figure 9-7. Phase Control with ground connection



To connect a mains switch to the integrated circuit, an optocoupled connection is the preferred option (see Figure 9-8).

Figure 9-8. Mains switch connection



Nevertheless, It's also possible to use a Zener diode to detect the switch too (see Figure 9-9).

Figure 9-9. Switch detection circuit using a Zener diode

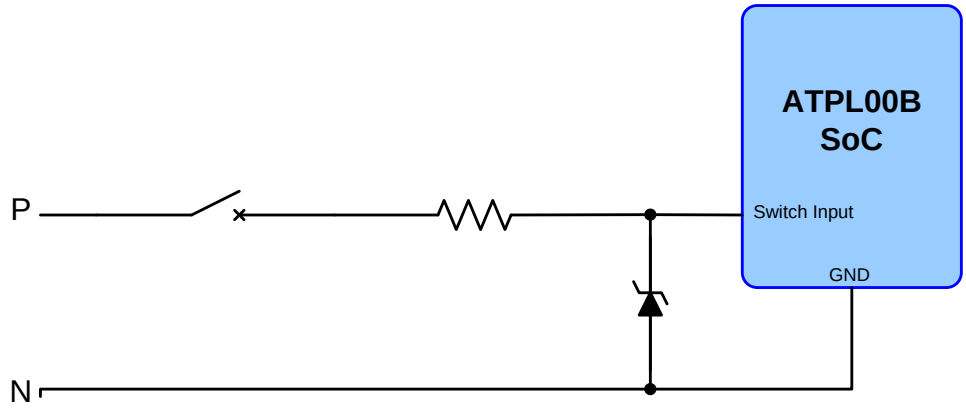
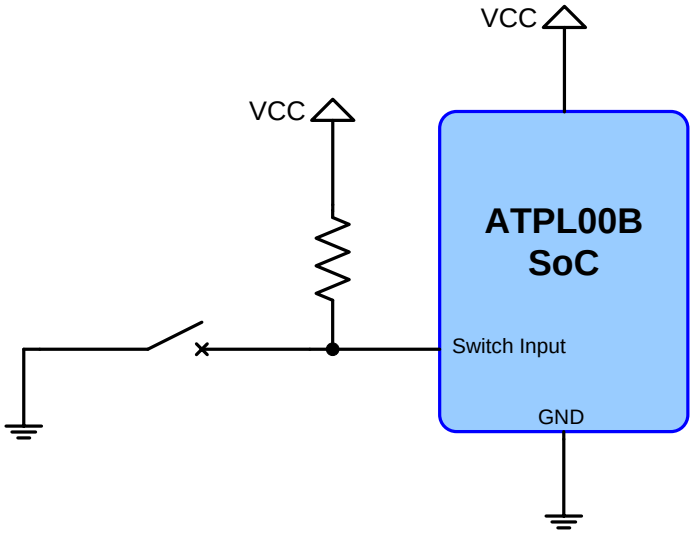


Figure 9-10 shows how to connect a low voltage switch to the integrated circuit.

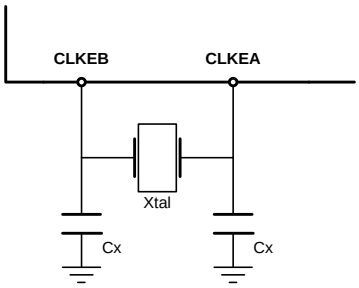
Figure 9-10. Direct current switch (Low Voltage switch)



Notes: 1. This pin is part of the JTAG Boundary Scan interface and is only used for boundary scan purposes

2. See supported devices section **8.3.2**

3. The crystal should be located as close as possible to CLKEA and CLKEB pins. Recommended value for Cx is 18pF. This value may depend on the specific crystal characteristics



4. Different configurations allowed depending on external topology and net behavior

10. Media Access Layer

In power line communication (PLC) systems, medium access control (MAC) tasks require a high percent of the CPU available computational time. To reduce the computational load of the integrated MCU (8051C3A Core) the ATPL00B accelerates the execution of critical tasks by means of additional specific hardware units such as ADD1210 hardwired MAC unit

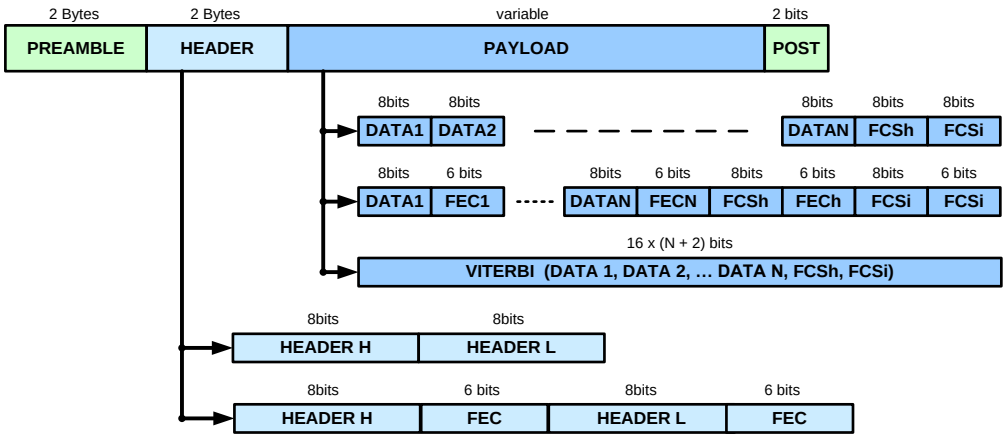
MAC functional capabilities involve the construction of message packets and the management of correction and error detection mechanisms.

The ATPL00B MAC is compatible with EHS and KONNEX. Moreover, its design is very versatile and allows the construction of a wide range of datagram structures with the only constrain of hardware correction and detection codes.

10.1 Packet Encapsulation

Depending on enabled error correction and detection mechanisms, MAC can encapsulate packets following different configurations:

Figure 10-1. Packet Encapsulation diagram



A standard packet encapsulation has the following structure

Figure 10-2. Datagram structure

2 byte	2 byte	variable	2 bit
PREAMBLE	HEADER	PAYLOAD	POSTAMBLE

- **Preamble:** two bytes long, containing 0xAAAA for bit synchronization. The preamble size is configurable by a field in CTRL register 10.3.1
- **Header:** two bytes long. It defines the type of datagram. ATPL00B supports FEC (Forward Error Correction) shielded and unshielded headers. In shielded headers, each byte of the header field is protected by a FEC field as shown below:
    - Unshielded header:

8-bit	8-bit
Header MSB	Header LSB

    - Shielded header:

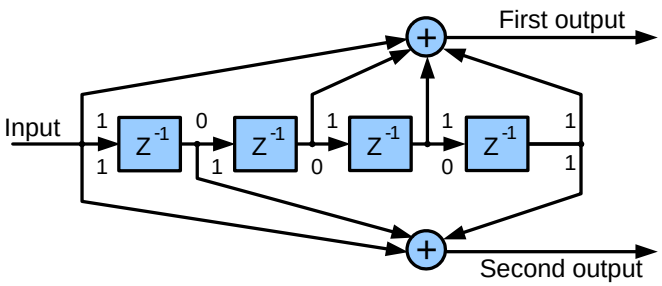
8-bit	6-bit	8-bit	6-bit
Header MSB	FEC	Header LSB	FEC

- **Payload:** This part of the message contains the data block. It also includes optional error correction information. ATPL00B supports Block Coding (FEC) and Convolutional Coding and a Frame Check Sequence (FCS) algorithm which is performed on the packet in order to detect additional errors

  The hardware-implemented FEC, with 8 bit of data and 6 bit of protection, is capable of correcting a 3 bit error burst in a 14 bit block. The generator polynomial is:

  The Viterbi algorithm is implemented with a constraint length (K) of 5, the code generator polynomials in octal are 27 and 31 and the decision method is soft-decision.

  Figure 10-3. Viterbi Encoder scheme



  A 16 bit interleaving block, implemented with a 4x4 square matrix, improves the Viterbi performance.

  A 2-byte long FCS helps to detect errors that could not be corrected by the FEC algorithm. The polynomial used to compute FCS is:

  In order to ensure proper error detection by the FCS, a maximum of 64 data bytes should be sent by frame.

  According to the correction mechanism selected, the MAC generates different types of payload structures:

  - Payload without protection, length=8*(n+2) bits

8-bit							8-bit	8-bit
DATA1	DATA2	DATA3	DATA4	DATA5	…	DATAn	FCSh	FCSl

  - Payload with FEC protection, length=14*(n+2) bits

8-bit	6-bit				8-bit	6-bit	8-bit	6-bit
DATA1	FEC1	…	DATAn	FECn	FCSh	FECh	FCSl	FECl

  - Payload with Viterbi protection, length=16*(n+2) bits

Viterbi (DATA1, DATA2, …, DATAn, FCSh, FCSl)

  -

- **Postamble:** This field is two bit long to allow full reception of significant data. Both bits are the complement of the last bit of the message.

10.2 MAC Architecture

The interface between the MCU and the MAC is achieved by addressing it as external data RAM. There are fifteen configuration registers in the MAC accessible to the MCU, mapped from 0xFE00 to 0xFE0E.

The tasks to perform and the MAC operation are controlled by the configuration registers settings, which are read by the MAC control logic. The MAC emitter-receiver logic sends or receives serial data through the modem, MSb first, at the selected baud rate.

Interrupt 0, located at INT0 internal port (P3.2), is triggered by the MAC whenever a new data byte is received. It can be configured to be triggered also in transmission (default value: INT0 is not triggered in transmission).

10.3 MAC Configuration Registers

Table 10-1. **MAC Configuration registers address maps**

Address	Register name	Reset Value (hex)
0xFE00	CTRL	0x4E
0xFE01	FLAGS1	0x10
0xFE02	FLAGS2	0x00
0xFE03	DATA	0x00
0xFE04	TH_PREAM	0x3F
0xFE05	TH_HEADER	0x00
0xFE06	BAUDRATE	0x02
0xFE07	TRACK	0x04
0xFE08	PREAM	0x02
0xFE09	CTRL2	0x00
0xFE0A	BC	0xFF
0xFE0B	MODE1	0x0C
0xFE0C	MODE2	0x0C
0xFE0D	MODE3	0x0C
0xFE0E	MODE4	0x0C
0xFE44	VTB_BE_HARD	0x00
0xFE45	VTB_BE_SOFTH	0x00
0xFE46	VTB_BE_SOFTL	0x00
0xFE47	FEC_BER	0x00

By default, reception mode has a baud rate of 2400 bps, the MAC computes the FEC and FCS fields, the TX interrupt is disabled and the header is unshielded. No preamble length restriction is needed in the header detection process.

10.3.1 CTRL register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
CTRL	A1	A0	BYTE	WHD	FCS	FEC	TXRX	STOP

Name: CTRL

Address: 0xFE00

Reset: “01001110”

The CTRL register holds mode operation bits and special purpose bits. Mode operation bits control the MAC behavior. The MCU has full access to this register.

Note: See description of BYTE, WHD, FEC, VTB and FCS modes in Operating modes section (10.4)

- **A(1:0):** Preamble Size.

  Bits A1 and A0 determine how many 0xA’s must precede the header field in order to consider the data sequence as a valid packet in reception. As the preamble size increases, the number of “false packets” received will decrease. The preamble is not protected against burst noise and a single bit error in this field will cause a packet loss.
  - “00”: Preamble=0xA
  - “01”: Preamble=0xAA
  - “10”: Preamble=0xAAA
  - “11”: Preamble=0xAAAA

  Note: This only concerns the reception of packets.

  Note: This field is not taken into account if ZERO_A field in FLAGS1 register (see 10.3.2) is enabled
- **BYTE:** Byte mode.

  This bit controls the BYTE mode operation when BYP and WHD bits are clear. In this mode, bytes are sent or received without protection mechanism.
  - ‘0’: BYTE mode disabled
  - ‘1’: BYTE mode enabled
- **WHD:** Write Headers mode

  This bit controls the Write Headers mode operation when BYP bit is clear. In this mode, the MCU configures the desired values of the four programmable headers. The hardware will only detect those packets whose headers coincide with one of the header models stored.
  - ‘0’: Write Headers mode disabled
  - ‘1’: Write Headers mode enabled
- **FCS:** Frame Check Sequence operation

  When this bit is set and BYTE (with ADV bit set, see CTR2 register in 10.3.10), FEC or VTB modes are also set, the FCS mode operation is active and the FCS field computation is assumed by the MAC. Clear this bit if FCS is not used or the FCS field computation is done by the MCU.
  - ‘0’: FCS operation disabled
  - ‘1’: FCS operation enabled
- **FEC:** Forward Error Correction operation

  If this bit is set and BYP, WHD, BYTE and VTB bits are clear, the FEC field computation is assumed by the MAC. Clear this bit if the packet doesn’t have FEC protection or the FEC field computation is done by the MCU.

- ‘0’: FEC operation disabled
- ’1’: FEC operation enabled

- **TXRX:** Transmission/Reception mode

  This bit selects if data is being transmitted/received to/from the modem
  - ‘0’: Transmission mode
  - ’1’: Reception mode
- **STOP:** This bit resets the MAC logic circuitry and aborts the current reception or transmission process. A “STOP cycle” (STOP bit must be set and then cleared) has to be issued by the MCU every time a packet reception or transmission is finished or being aborted. The STOP cycle doesn’t reset the configuration registers.
  - ‘0’: MAC is in normal operation mode
  - ’1’: MAC is in reset state

10.3.2 FLAGS1 register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
FLAGS1	BYP	--	--	ZER0_A	NMDR	RxFEC	RX	TX

Name: FLAGS1

Address: 0xFE01

Reset: “00010000”

This register holds a mode operation bit, some special purpose bits and the RX and TX flags to control the transmission and reception processes. The MCU has full access to this register.

- **--:** Reserved bit
- **BYP:** Bypass mode

  When this mode is enabled, the MAC logic is disabled and the MCU takes full control of the PLC modem
  - ‘0’: Bypass mode disabled
  - ‘1’: Bypass mode enabled
- **ZER0_A:** Errors allowed in preamble

  This field selects whether erroneous bits are allowed in preamble reception or not
  - ‘0’: No errors are allowed in preamble reception. A(1:0) field (see CTRL register in 10.3.1) sets the number of 0xA’s expected to be received. If there is any erroneous bit in this preamble reception, the frame will be discarded.
  - ‘1’: Errors are allowed in preamble reception. The frame will be discarded only if the erroneous bits received in preamble exceed a threshold (see TH_PREAM register in 10.3.5).
- **NMDR:** No more data required

  This bit should be set by the MCU after the last byte of the payload is being sent or received through the DATA register. In FCS mode it would mean that the next field to be processed by the MAC is the FCS field.

  Setting the NMDR bit is only required in the following situations:

  a) Message transmissions with TX interrupt enabled.

  b) Message receptions in FCS mode with Viterbi algorithm disabled.
  - ‘0’: NMDR deasserted
  - ‘1’: NMDR asserted
- **RxFEC:** Receive FEC fields

  If this bit is set, FEC fields are also delivered by the MAC to the MCU by means of DATA register. If this bit is clear, the MAC delivers to the MCU only the data bytes. See DATA register section and “Operation Modes” chapter for more information about how the FEC bits are made accessible to the MCU.
  - ‘0’: Only data bytes are read by the MCU
  - ’1’: Both data and FEC are read by the MCU

- **RX:** Reception flag

  This flag is set by the MAC when a new byte is received, which automatically generates an external interrupt (INT0) that alerts the MCU. The RX flag should be cleared by the MCU after the DATA register value has been read

  Rx bit is also set to ‘1’ in transmission every time a FCS-field byte is sent. Thus, the MAC alerts the MCU to read and store the value of the FCS (sent by the MAC), which is later needed to receive and validate the correspondent ACK

- **TX:** **Transmission flag**

  The MCU has to set this flag after writing a new byte in the DATA register for transmission. The MAC will clear this bit when it is ready to accept more data. The MCU should only write into the DATA register when the TX bit is clear.

10.3.3 FLAGS2 register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
FLAGS2	HDEC	HD_SH	TH_MODE	END_TX	HD1	HD0	EFCS	EFEC

Name: FLAGS2

Address: 0xFE02

Reset: “00000000”

This register holds three configuration bits and five read-only information bits (FLAGS2(4:0)).

- **HDEC:** Header decoding mode

  When the MAC is configured to receive unshielded headers (HD_SH=’0’), this bit selects the way the errors are calculated when a header is received.
  - ‘0’: Soft decision. In soft decision there are eight levels of decision. A “strong 0” is represented with a value of “0”, while a “strong 1” is represented with a value of “7”. The rest of values are intermediate, so “3” is used to represent a “weak 0” and “4” represents a “weak 1”. Soft decision calculates the error in one bit received as the distance in decision levels between the value received (a value between “0” and “7”) and the corrected one (“0” or “7”).
  - ‘1’: Hard decision. In hard decision there are only two decision levels. If the received value is different than the corrected one, the error value taken is “7”. Otherwise, the error value taken is “0”.
- **HD_SH:** Shielded header

  When this bit is set, shielded header packets are in use. A shielded header packet means that the header is protected by FEC fields. Shielded headers are not compatible with EHS-KONNEX.

  This bit is only taken into account in headers reception. This bit does not affect transmission.
  - ‘0’: Shielded headers disable
  - ‘1’: Shielded headers enable
- **TH_MODE:** Header threshold mode

  When the MAC is configured to receive unshielded headers (HD_SH=’0’), this bit configures how the header detection thresholds are used.
  - ‘0’: Errors found in preamble and errors found in header are not accumulated. Threshold for errors in preamble is stored in TH_PREAM register (see 10.3.5) while threshold for errors in header is stored in TH_HEADER register (see 10.3.6).
  - ‘1’: Errors found in preamble and errors found in header are accumulated. The threshold for this addition is stored in TH_HEADER register.
- **END_TX:** End of transmission flag

  This flag is set by the MAC when the current transmission finishes. A STOP cycle clears this bit.
  - ‘0’: Transmission in curse
  - ‘1’: End of transmission

- **HD(1:0):** Header number

  When the MAC detects one of the programmed header values in the incoming data sequence, and once the first byte following the header has been received, the MAC triggers the external interruption. The data byte will be stored in DATA register, and the header type will be stored in HD(1:0). That is, the MAC doesn't send the header bytes to the MCU through DATA register.
  - "00": Header 1 (default value = 0x9B58)
  - "01": Header 2 (default value = 0xE958)
  - "10": Header 3 (default value = 0x24E7)
  - "11": Header 4 (default value = 0xA5D8)
- **EFCS:** FCS error flag

  This flag is set if an error in the FCS field is detected in the incoming packet.
  - '0': No error detected
  - '1': Error detected
- **EFEC:** FEC error flag

  This flag is set if an error in the FEC field is detected in the incoming packet.
  - '0': No error detected
  - '1': Error detected

10.3.4 DATA register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
DATA	D7	D6	D5	D4	D3	D2	D1	D0

Name: DATA

Address: 0xFE03

Reset: “00000000”

In default mode, DATA register holds the byte value to be sent in transmission or the last received byte in reception. This register is also used to program the header values (see WHD mode section for more details). The MCU has full access to this register.

In some operation modes, the FEC field is also transmitted by means of DATA register. As the FEC field is 6-bit long, it is stored in the 6 lsb (less significant bits) in DATA register according to one of the configurations below.

- DATA register in FEC mode

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
DATA	--	--	FEC5	FEC4	FEC3	FEC2	FEC1	FEC0

In FEC mode, when RxFEC=’1’, the MAC also transfers the FEC field to the MCU in reception (if RxFEC=’0’, only data bytes will be transferred), and this is done according to configuration shown above

- DATA register in NULL mode (see 10.4.8)

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
DATA	--	--	/FEC5	/FEC4	/FEC3	/FEC2	/FEC1	/FEC0

On the other hand, when MAC is in NULL mode (that is, the MAC does not manage the FEC field), the MCU must manage FEC fields following the configuration shown above. Due to not-MAC intervention, the MCU will receive the FEC inverted, so the FEC must be also written inverted in the DATA register by the MCU when transmitting.

10.3.5 TH_PREAM register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TH_PREAM	--	--	D5	D4	D3	D2	D1	D0

Name: TH_PREAM

Address: 0xFE04

Reset: “00111111”

Preamble threshold register

- **D(5:0):** Preamble threshold.

  This field stores an upper threshold for errors detected in preamble when receiving unshielded headers (header without FEC, HD_SH=’0’). If this threshold is exceeded, the frame will be discarded.

  Note: This register is taken into account only if TH_MODE=’0’.

  Note: This register is taken into account only if ZERO_A=’1’.

10.3.6 TH_HEADER register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TH_HEADER	D7	D6	D5	D4	D3	D2	D1	D0

Name: TH_HEADER

Address: 0xFE05

Reset: “00000000”

Header threshold register

- **D(7:0):** Preamble threshold.

  This register stores an upper threshold for errors detected in the header, when receiving unshielded headers (HD_SH=’0’). The received header is compared with the four headers stored in memory and error value is calculated. If the error value is equal or below the value in TH_HEADER, the header is taken as valid.

  If this threshold is exceeded, the frame will be discarded.

  Note: If TH_MODE field in FLAGS2 register is set, TH_HEADER register stores the threshold for errors found in header + errors found in preamble.

10.3.7 BAUDRATE register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
BAUDRATE	--	--	--	--	--	--	BR1	BR0

Name: BAUDRATE

Address: 0xFE06

Reset: “00000010”

The MCU can configure the baud rate of the system by writing to the BAUDRATE register. The MCU has full access to this register.

- **BR(1:0):** This field sets the baud rate.
  - “00”: 600 bps
  - “01”: 1200 bps
  - “10”: 2400 bps
  - “11”: 4800 bps

10.3.8 TRACK register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TRACK	--	D6	D5	D4	D3	D2	D1	D0

Name: TRACK

Address: 0xFE07

Reset: “00000100”

This register is used for internal debug. Reset value is the suitable for correct operation.

This register **must not** be modified, unless otherwise stated.

10.3.9 PREAM register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
PREAM	--	D6	D5	D4	D3	D2	D1	D0

Name: PREAM

Address: 0xFE08

Reset: “00000010”

This register is used for internal debug. Reset value is the suitable one for correct operation.

This register **must not** be modified, unless otherwise stated.

10.3.10 **CTRL2 register**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
CTRL2	--	--	--	--	TXIE	--	ADV	VTB

Name: CTRL2

Address: 0xFE09

Reset: “00000000”

This register holds configuration bits that control new features of the ATPL00B that are not backwards compatible.

- **--:** Reserved bit
- **TXIE:** This bit enables the transmission interrupt INT0 in operation modes that are not WHD
  - ‘0’: TX interrupt disabled
  - ‘1’: TX interrupt enabled
- **ADV:** Advanced mode

  Changes the behavior of BYTE and FCS operation mode bits, allowing the combined BYTE + FCS mode. This mode is not backward compatible
  - ‘0’: Advanced mode disabled
  - ‘1’: Advanced mode enabled
- **VTB:** VTB mode

  Activate VTB mode when BYP, WHD and BYTE bits are clear
  - ‘0’: VTB mode disabled
  - ‘1’: VTB mode enabled

10.3.11 **BC register**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
BC	D7	D6	D5	D4	D3	D2	D1	D0

Name: BC

Address: 0xFE0A

Reset: “11111111”

- **D(7:0):** The BC register must be configured by the MCU in a reception process when Viterbi mode is enabled.

  The BC register holds the byte count parameter, that is, the total number of bytes protected by the Viterbi algorithm. Once the MCU knows the total length of the frame that is receiving, it must configure the BC register, which means this register must be dynamically updated with every received frame.

  When Viterbi algorithm enabled, the byte count parameter must be decoded before the execution of the flushing stage. As a consequence, the optimum position of the byte count parameter is the closest to the header section. The MCU has full access to this register.

10.3.12 **MODE1 to MODE4 registers**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
MODE4	--	--	--	VTBi	FECi	FCSi	BYTEi	ENi

Name: MODE1 – MODE4

Address: 0xFE0B - 0xFE0E

Reset: “00001100”, 00001100”, 00001100”, 00001100”.

MODEi registers allow associating a MAC operation mode in reception to each of the four programmable headers stored in memory. This way, if mode-i (where i=1,2,3,4) is enabled (ENi=’1’), when the MAC receives the header-i, it automatically processes this frame according to the operation mode defined by MODEi.

VTBi, FECi,FCSi and BYTEi fields work in the same way as the corresponding fields in the general configuration: VTB, FEC, FCS and BYTE respectively.

These registers are only applicable to frame reception. Transmission mode is still controlled by the general configuration bits (VTB, FEC, FCS, BYTE).

10.3.13 **VTB_BE_HARD register**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
VTB_BE_HARD	D7	D6	D5	D4	D3	D2	D1	D0

Name: VTB_BE_HARD

Address: 0xFE44

Reset: “00000000”

- **D(7:0):** Stores the number of errors accumulated in the last received frame using Viterbi hard.

  The register holds the value until the next frame is completely received, and then the value is updated.

  Note: The value in this register is only correct if the message has been correctly received (successful CRC check).

10.3.14 **VTB_BE_SOFT registers**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
VTB_BE_SOFTH	D15	D14	D13	D12	D11	D10	D9	D8
VTB_BE_SOFTL	D7	D6	D5	D4	D3	D2	D1	D0

Name: VTB_BE_SOFTH – VTB_BE_SOFTL

Address: 0xFE45 - 0xFE46

Reset: “00000000”, “00000000”

- **D(15:0):** Stores the number of errors accumulated in the last received frame using Viterbi soft.

  The register holds the value until the next frame is completely received, and then the value is updated.

  Note: The value in this register is correct only if the message has been correctly received (successful CRC check).

10.3.15 **FEC_BER register**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
FEC_BER	D7	D6	D5	D4	D3	D2	D1	D0

Name: FEC_BER

Address: 0xFE47

Reset: “00000000”

- **D(7:0):** Stores the number of errors accumulated in a frame reception using Forward Error Correction.

10.4 Operation modes

The different operation modes are summarized in Table 10-2

Table 10-2. **MAC Operation modes**

[FEC, FCS, VTB]	[BYP, WHD, BYTE] 000	001	01X	1XX
0X0	NULL	BYTE	WHD	BYPASS
X01	VTB	BYTE	WHD	BYPASS
X11	VTB + FCS	BYTE	WHD	BYPASS
100	FEC	BYTE	WHD	BYPASS
110	FEC + FCS	BYTE	WHD	BYPASS

As mentioned before, if bit ADV is set, a special mode BYTE+FCS is available. This mode is selected when BYTE and FCS bits are set and BYP and WHD bits are cleared.

10.4.2 BYPASS mode

In BYPASS mode, the MCU takes control of the ATPL00B modem. The MAC circuitry is disabled and the MAC layer has to be fully implemented by software.

The following table shows the connections between the MCU and the modem in bypass mode operation:

Table 10-3. **MCU-modem connection in bypass mode**

MCU	Modem
P1.0	CD
P1.2	TxRx
P1.1	TxD
P3.2	RxD

10.4.3 WHD mode

In WHD mode it is possible to change the value of the programmable headers.

The MAC detects how many headers have been programmed and uses them to look for valid incoming packets during reception. Up to four headers can be programmed in this mode. See WHD mode operation procedure in 10.5.2for detailed information.

10.4.4 BYTE mode

In BYTE mode, the MAC logic transfers data to and from the modem in byte format. Thus, transferred bytes do not have protection field. This mode should be used by the MCU to send PREAMBLE and HEADER fields in EHS-KONNEX compatible packets.

10.4.5 FEC mode

FEC is one of the two available error correction mechanisms. In this mode, the MAC appends a FEC field to each data byte during transmission and checks its value during reception.

10.4.6 VTB mode

The second error correction mechanism is VTB. In VTB mode, the MAC uses convolutional encoding to encode data bytes during transmission. During reception, the data bytes are decoded applying the Viterbi algorithm if VTB mode is chosen.

10.4.7 FCS mode

In FCS mode, a FCS field is appended to the packet during transmission, and checked afterwards during reception. This mode has to be used together with any of the following modes: BYTE (with ADV set), FEC or VTB.

10.4.8 NULL mode

In NULL mode, the MAC logic assumes that FEC fields are being used in the data packet but computed by the MCU. As a consequence, each data byte is followed by a FEC field, and only the 6 bits of the DATA register that holds the FEC field are processed.

10.5 Operation Procedures

This section describes the different operation procedures that the MCU has to take in order to handle with the MAC hardware. Except for the STOP cycle which is common to all modes, the rest of procedures are mode dependent.

10.5.1 STOP cycle

A MAC STOP cycle resets the MAC logic circuitry and it can also abort the current transmission or reception process. The MCU must generate a STOP cycle every time a packet transmission or reception is completed or aborted.

10.5.2 WHD mode operation procedure

The WHD mode operation procedure allows the MCU to change the default values of the programmable headers.

The default number of programmable headers involved in the header detection process is 4. However, if a header programming procedure takes place, this number will be equal to the number of programmed headers.

The four header values are stored in an eight byte stack area. The value of FLAGS2[HD[1:0]] is used by the MAC logic to point into the stack to the header that is being received in the incoming message.

In WHD mode there is no data sent to the modem, but the CTRL[TXRX] bit has to be cleared anyway. Additionally, the TX interrupt is disabled independently of the CTR2[TXIE] bit value.

Example: Header programming procedure

An example of the header programming procedure is provided in this section. The purpose is to change the header values the following way:

Default values

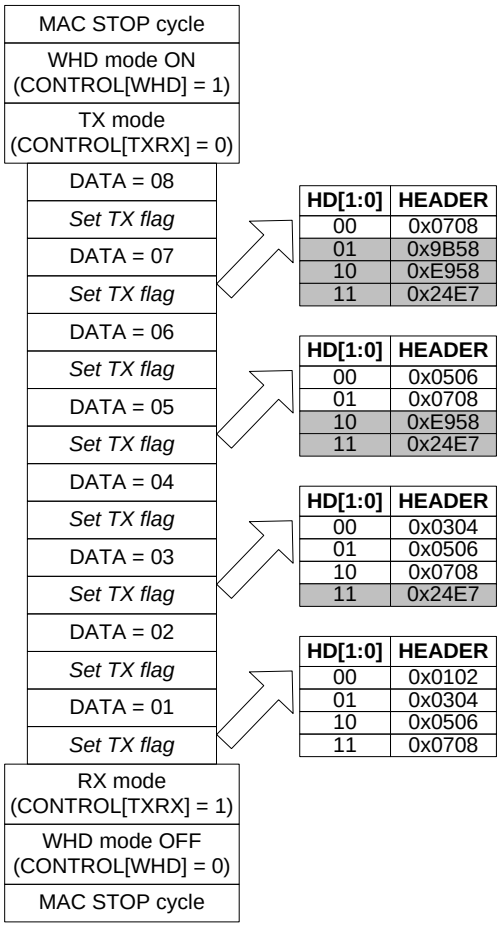
HD[1:0]	HEADER
00	0x9B58
01	0xE958
10	0x24E7
11	0xA5D8

Target values

HD[1:0]	HEADER
00	0x0102
01	0x0304
10	0x0506
11	0x0708

The procedure to program these values is shown in Figure 10-4 Note that the shadowed headers would not be involved in the header detection process.

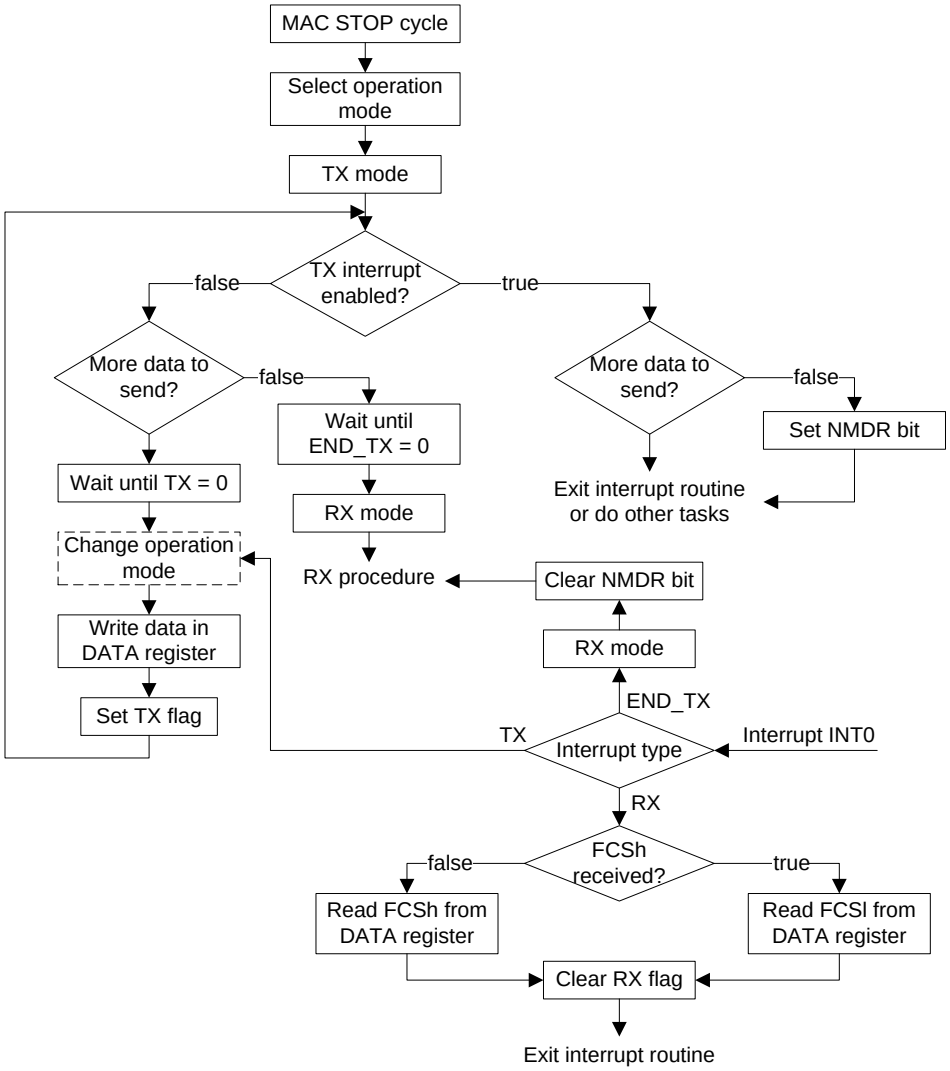
Figure 10-4. WHD procedure example



10.5.3 Transmission procedure

To send a PLC message, the MCU must follow the transmission procedure outlined in Figure 10-5. The described procedure is independent of the selected operation mode and can be executed with TX interrupt enabled or disabled. The carrier detect signal (P10) must be checked by the MCU before executing the transmission procedure.

Figure 10-5. Transmission procedure flowchart



At the beginning of the procedure, the MCU must configure the MAC in transmission mode and select the desired operation mode.

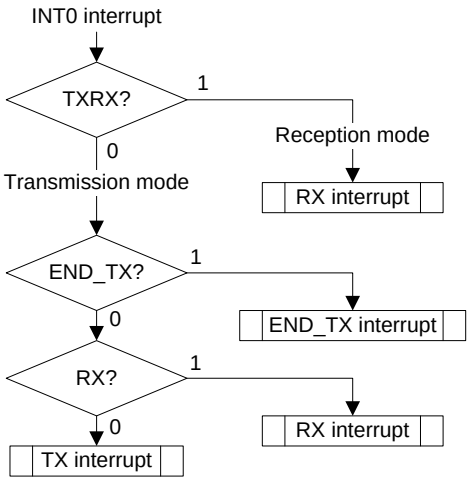
The decision ‘More data to send?’ returns FALSE when there are no more bytes to send in modes without FCS. When FCS is used, the next byte is the highest byte of the FCS field.

The process ‘Change operation mode’ is optional. When used, it is possible to change the operation mode to compose messages with a complex structure. For example, in a EHS-KONNEX message, the preamble and header fields are sent in BYTE mode without protection and the rest of the message in FEC + FCS mode.

During transmission, there are three types of INT0 interrupts: TX, RX and END_TX. Interrupts of type TX and END_TX are only active when bit CTR2[TXIE] is set. RX type interrupt is active when CTRL[FCS] is set. The TX type interrupt alerts the MCU whenever the MAC is ready to accept a new byte to be sent. The RX type interrupt, in transmission mode, is used to pass the FCS value from the MAC to the MCU. Finally, the END_TX type interrupt indicates that the last bit of the POSTAMBLE field was sent to the modem, that is, the message is finished.

To know what type of INT0 interrupt has occurred, the MCU has to check the CTRL[TXRX] bit and flags FLAGS1[RX] and FLAGS2[END_TX]. Then, by following the algorithm shown in Figure 10-6, it can determine the appropriate interrupt handler.

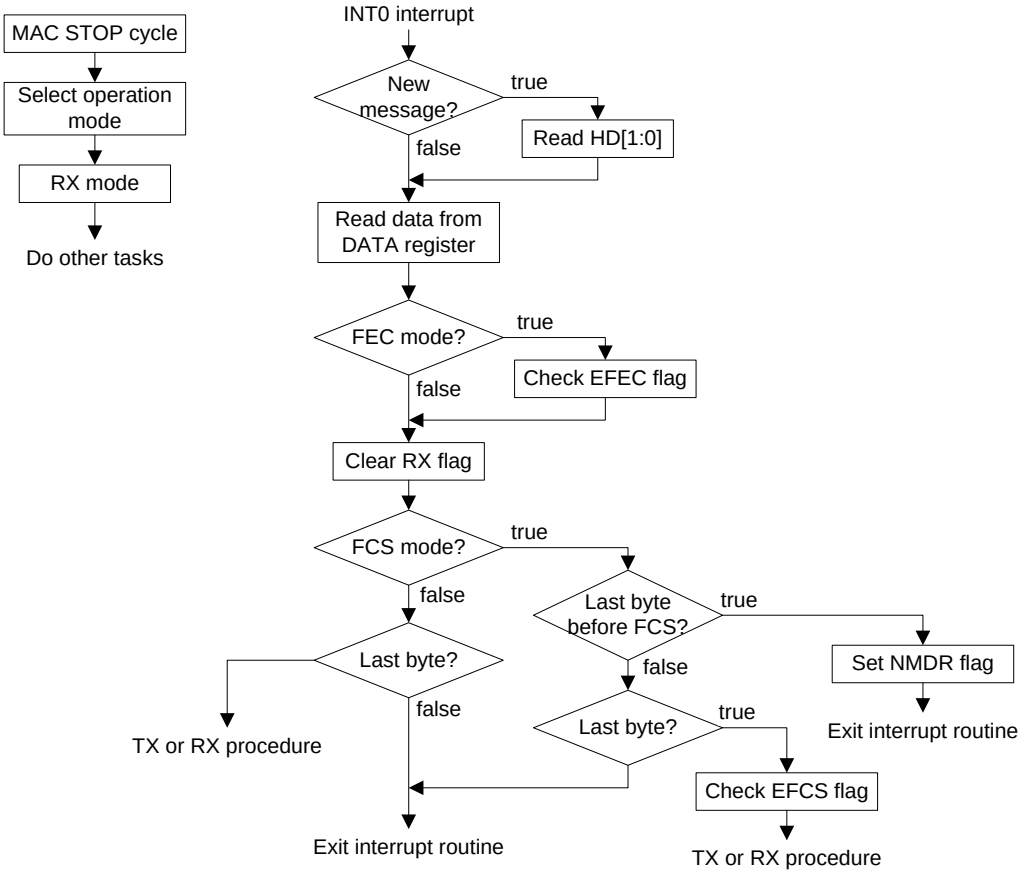
Figure 10-6. Interrupt type detection



10.5.4 Reception procedure

After configuring the system in reception mode, the MAC looks for a valid header in the incoming data sequence. When a valid header is detected, the following byte received is left in the DATA register, bits HD[1:0] are set according to the header detected and MAC raises the INT0 interrupt to alert the MCU.

Figure 10-7. Reception procedure flowchart



For each processed byte, an interrupt alerts the MCU. If the FLAGS1[RxFEC] bit is set, then FEC fields are also passed to the MCU via the interrupt method. If the message includes error protection and/or detection mechanism, the MAC hardware uses the flags FLAGS2[EFEC] and FLAG2[EFCS] to inform about the computation results. However, it is a MCU decision to abort the reception process or not. The reception of the message can be aborted anytime by issuing a STOP cycle.

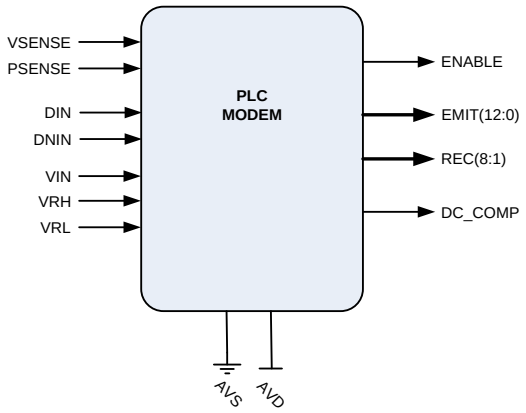
11. PLC Modem

The ATPL00B is a Power Line Communications System on Chip. It includes an enhanced 8051 microcontroller (IP core ADD8051C3A), a Media Access Controller (MAC) (IP core ADD1210) and a PLC Modem circuit for the EHS/KNX Power Line medium specifications (IP core ADD1310).

The ATPL00B is a multi Baud rate device, which can work at 600 bauds, 1200 bauds, 2400 bauds or 4800 bauds. ATPL00B PLC modem works at a 132.5 kHz carrier frequency.

The PLC modem can be controlled using the hardwired MAC ADD1210 or it can be controlled by the microcontroller ADD8051C3A, nonetheless when the microcontroller takes control the MAC is still used, but in a bypass mode.

Figure 11-1. PLC Modem diagram



The modem has a set of configuration registers (listed in Table 11-5) which are mapped in the external data space of the microcontroller. Many bits in these registers are not user definable, and the user **must not** write to these bits.

In order to have direct access from the microcontroller to the modem, its control signals (TX,RX,TX/RX,CD) are connected to microcontroller ports. The port P3.2 has a double function. When the MAC is in full operation mode, P3.2 is used to interrupt the microcontroller when a new byte, or a new byte+FEC is received. When the MAC is in bypass mode the reception output of the modem, RX, is connected to P3.2

Table 11-1. **MUC-MAC connection**

MCU	MAC	
	bypass	Full operation
P3.2	Rx	Rx Interrupt
P1.0	CD	CD
P1.1	Tx	--
P1.2	TxRx	--

When the MAC is used in full operation mode all the modem control signals (TX,RX,TX/RX,CD) are connected only to the MAC.

11.1 Frequency coding

Logic symbol ‘1’ is sent using a frequency value slightly below Fcarrier while the upper side is used to represent the logic symbol ‘0’, according to:

- F(“1”) = Fcarrier - baud rate/2
- F(“0”) = Fcarrier + baud rate /2

Besides there is another option, where you can choose (frequency deviation=0.5):

- F(“1”) = Fcarrier - baud rate/4
- F(“0”) = Fcarrier + baud rate/4

Exact frequencies for every baud rate can be found in 0.

Note: 0.5 deviation value is not allowed when working at Baud Rate = 600. See Table 11-2 for suitable values.

Table 11-2. **Allowed deviation values by baud rate**

Baud rate	F (Hz)	Deviation
600[(1)]	-- 600	-- 1
1200	600 1200	0.5 1
2400[(2)]	1200 2400[(2)]	0.5 1
4800	2400 4800	0.5 1

Note: 1.Deviation value = 0.5 not allowed

Note: 2.Default value

A couple of modems have been implemented inside ATPL00B (each one with Tx and Rx capabilities), so we can select between them to establish different values or frequency deviation for transmission and reception (See DEV_TX and DEV_RX fields in CONFIG register).

This allows the device to be configured to comply with KONNEX requirements, setting a deviation value = 1 for reception and a deviation value = 0.5 for transmission.

Table 11-3. **Allowed Tx-Rx deviation values**

Tx	Rx
[(1)]0.5	[(1)]0.5
0.5	1
1	1

Note: 1. If a reception deviation value = 0.5 is selected, same transmission deviation value = 0.5 must be selected as well. In this case, only baud rate=2400bauds can be used

Table 11-4. **Modem mark and space frequencies**

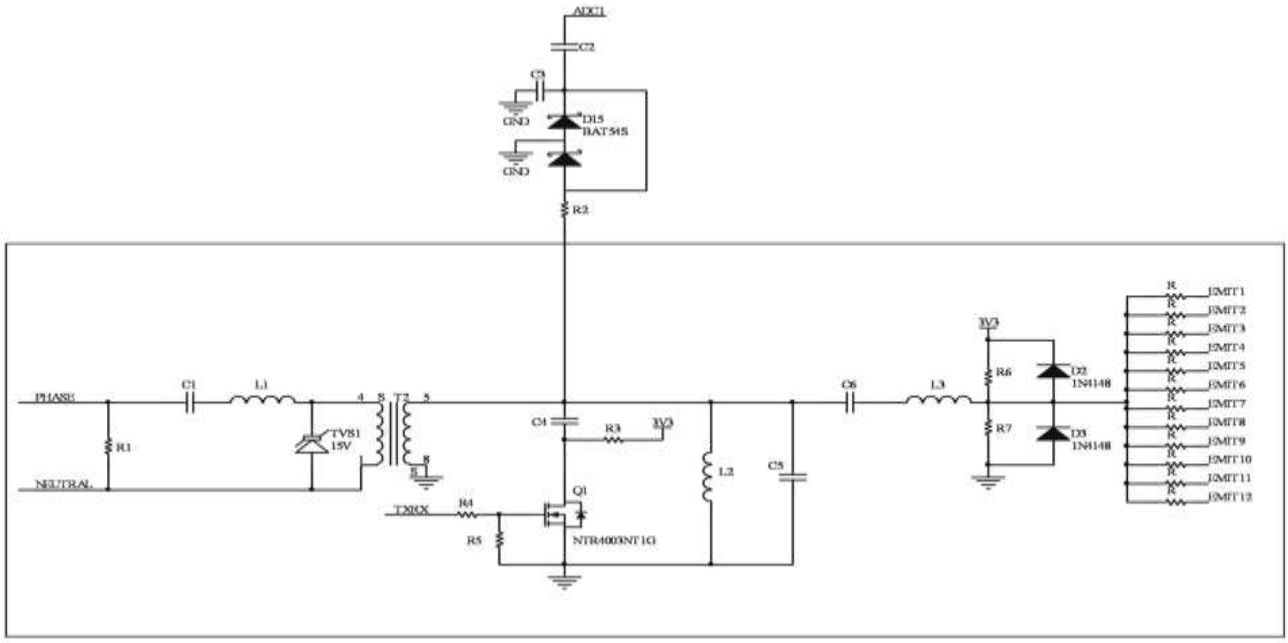
Carrier frequency (KHz)	Baud rate	Deviation	Exact frequency (Hz)	
			"1"	"0"
132,5	600	--	--	--
		1	132200	132800
	1200	0,5	132200	132800
		1	131900	133100
	2400	0,5	131900	133100
		1	131300	133700
	4800	0,5	131300	133700
		1	130100	134900

11.2 Modem Transmission and Reception

11.2.1 Transmission characteristics

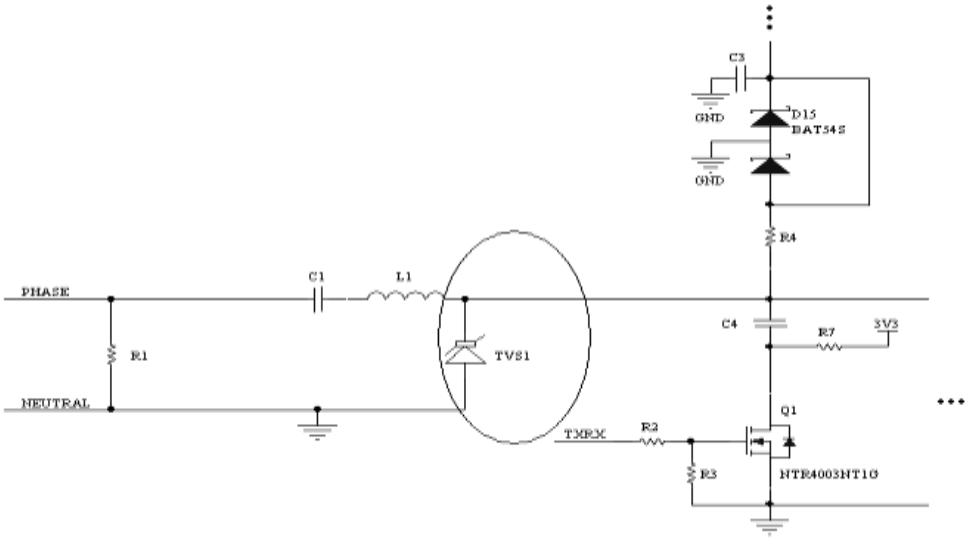
ATPL00B PLC Modem implements a direct transmission configuration, where the transmitted signal is generated by means of a resistor array. The value of the resistors determines the intensity that will be injected to the power line, and then the signal level. This low-cost and low-consumption configuration is intended for Home Automation purposes. An example of a possible application diagram is shown in Figure 11-2

Figure 11-2. Example of an emission circuit using a resistor array



Note: PLC modem also works properly with direct coupling, allowing final user to save transformer costs and reducing design size.

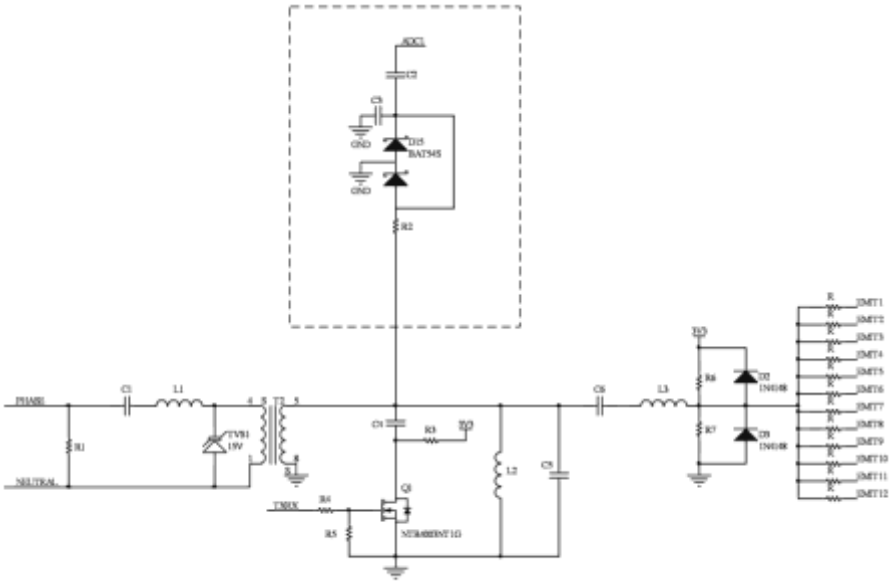
Figure 11-3. Direct coupling without transformer

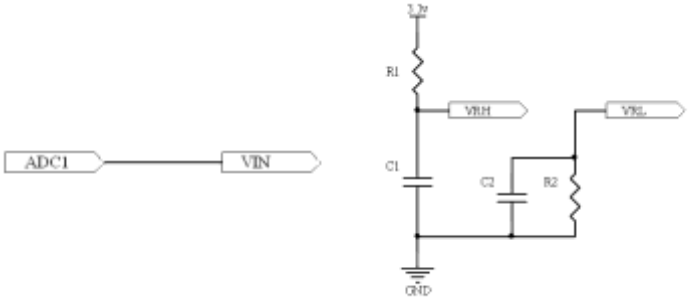


11.2.2 Reception characteristics

The receiving characteristics, input sensitivity and input impedance, are greatly influenced by the receiving circuitry. ATPL00B provides a Low-Cost Direct-Analog input stage using VIN, VRH and VRL inputs, allowing to save costs in external circuitry and providing robust performance.

Figure 11-4. Low Cost reception circuit





Note: ATPL00B also provides a High-Performance External-Comparator reception interface that allows the user to work with external comparator circuitry when looking for an extremely-high-temperature-environment working system. Please contact Atmel for further information.

11.3 PLC Modem Configuration registers

The modem has a set of configuration registers which are mapped in the external data space of the microcontroller and allow the user to set main working characteristics.

Table 11-5. **MODEM Registers map**

Address	Register name	Reset Value (hex)
0xFE14	GAIN	0x40
0xFE15	CONTROL2	0x80
0xFE17	CONFIG	0x06
0xFE18	GFSK_ADDR	0x00
0xFE19	GFSK_D3	0x00
0xFE1A	GFSK_D2	0x00
0xFE1B	GFSK_D1	0x00
0xFE1C	GFSK_D0	0x00
0xFE1F	COMP	0x04
0xFE40	KNX_EN	0x57
0xFE48	CD_ENABLE	0x01

Address	Register name	Reset Value (hex)
0xFE49	CD_DECISION_TIME	0x20
0xFE4A	CD_DECISION_ERROR	0x0F
0xFE4B	TXRX_CTL	0x00
0xFE4C	R1	0x60
0xFE4D	R2	0x60
0xFE4E	R3	0x60
0xFE4F	R4	0x60
0xFE50	R5	0xFF
0xFE51	R6	0xFF
0xFE52	R7	0xFF
0xFE53	R8	0xFF

11.3.1 GAIN register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
GAIN	G7	G6	G5	G4	G3	G2	G1	G0

Name: GAIN

Address: 0xFE14

Reset: “01000000”

This register controls the Amplitude of the emitted PLC signal

- **G(7:0):** Controls the Amplitude of the emitted PLC signal. Square output waveform is selected when the value is above 0x7B in order to minimize distortion
  - Values from 0x00 to 0x7B: Sine output waveform. The amplitude(*) varies according to:

    Where Vcc is the voltage supplied to the output analog stage (which varies depending on its configuration)
  - Values from 0x7C to 0xFF: Square output waveform. Any value above 0x7C will be automatically stored as 0x7C.

  Note: Amplitude values depend on the net impedance. The net impedance varies constantly.

11.3.2 CONTROL2 register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
CONTROL2	C_PD	--	--	--	--	--	--	STOP_MODEM

Name: CONTROL2

Address: 0xFE15

Reset: “10000000”

- **C_PD:** Internal Comparator Power Down
  - ‘0’: Normal state
  - ‘1’: Power down state
- **STOP_MODEM:** PLC Modem power down
  - ‘0’: Normal state
  - ‘1’: Power down state

11.3.3 CONFIG register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
CONFIG	KNX	DEV_RX	DEV_TX	--	--	EOP_EN	--	GFSK_EN

Name: CONFIG

Address: 0xFE17

Reset: “00000110”

- **KNX:** KONNEX normative

  If KNX=’0’, KONNEX normative is not complied, but a more solid transmission is achieved (the more A’s you add to the preamble, the better it works)

  In order to comply KONNEX normative **(*)**, this bit must be set.
  - ‘0’: KONNEX normative not compliant
  - ‘1’: KONNEX normative compliant

  (*) In order to comply with KONNEX normative, the following values must be used:
    - Baud rate = 2400 bauds
    - CONFIG(KNX) = ‘1’
    - CONFIG(DEV_TX) = ‘1’ (frequency deviation = 0.5)
    - CONFIG(DEV_RX) = ‘0’ (frequency deviation = 1)
    - KNX_EN register = 0x77
- **DEV_RX:** Deviation value in reception

  DEV_RX=’0’ is the normal mode of transmission which has a deviation value of 1. In case the user wants to work with a deviation value of 0.5, this bit must be set.
  - ‘0’: Frequency deviation=1
  - ‘1’: Frequency deviation=0.5

  Note: Frequency deviation=0.5 can only be used when working at 2400 bauds. Nevertheless, it is recommended to work with a frequency value deviation=1.
- **DEV_TX:** Deviation value in transmission

  DEV_TX=’0’ is the normal mode of transmission which has a deviation value of 1. In case the user wants to work with a deviation value of 0.5, this bit must be set.
  - ‘0’: Frequency deviation=1
  - ‘1’: Frequency deviation=0.5
- **EOP_EN:** Emission Output Pins Enable

  Enables the emission outputs (EMIT(1:12) pins).
  - 0 - Transmission outputs enabled
  - 1 - Transmission outputs disabled

  Note: This bit **must** be cleared (EOP_EN=’0’) before starting transmission in order to protect external circuitry.
- **GFSK_EN:** Gaussian output filter

  If set, configures the ADD1310 to use the Gaussian output filter, that smoothes pulses to limit its spectral width.

11.3.4 GFSK_ADDR register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
GFSK_ADDR	--	A6	A5	A4	A3	A2	A1	A0

Name: GFSK_ADDR

Address: 0xFE18

Reset: “00000000”

- **--:** Reserved bit
- **A(6:0):** This field contains 7 bits to address the table where Gaussian Filter predefined values are written in memory.

11.3.5 GFSK_DX registers

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
GFSK_D3	--	--	--	--	D27	D26	D25	D24
GFSK_D2	D23	D22	D21	D20	D19	D18	D17	D16
GFSK_D1	D15	D14	D13	D12	D11	D10	D9	D8
GFSK_D0	D7	D6	D5	D4	D3	D2	D1	D0

Name: GFSK_D3 – GFSK_D0

Address: 0xFE19 – 0xFE1C

Reset: “00000000”, “00000000”, “00000000”, “00000000”.

- **--:** Reserved bit
- **D(27:0):** The 28-bit value from the table address pointed by GFSK_ADDR register is stored in four registers named GFSK_D3 to GFSK_D0.

  It is also possible to write a desired data in these registers, and it will be automatically kept in the memory ADDRESS read from GFSK_ADDR register once the new data has been written. This is useful in case of changing the baud rate and a new table must be saved in memory before applying the filter.

11.3.6 COMP register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
COMP	--	--	--	--	--	--	--	CCLK_IDLE

Name: COMP

Address: 0xFE1F

Reset: “00000100”

- **--:** Reserved bit
- **CCLK_IDLE:** Converter Clock Idle

  Set the state of the clock that drives the external comparator (ENABLE pin)
  - ‘0’:External Comparator clock enabled
  - ‘1’: External Comparator clock disabled

11.3.7 KNX_EN register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
KNX_EN	--	D6	D5	D4	D3	D2	D1	D0

Name: KNX_EN

Address: 0xFE40

Reset: “01010111”

Note: This register is used to comply with KONNEX normative.

If a KONNEX normative compliant modem is required, D5 bit field must be set, thus KNX_EN=0x77 and values described in CONFIG register section must be used.

Else, default value **(KNX_EN=0x57)** is recommended.

11.3.8 CD_ENABLE register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
CD_ENABLE	--	--	--	--	--	--	--	EN

Name: CD_ENABLE

Address: 0xFE48

Reset: “00000001”

This register is used to enable/disable the automatic carrier detection

- **--:** Reserved bit
- **EN:** Automatic Carrier Detection enable
  - ‘0’: Carrier Detection depends on the value stored in THRESHOLD registers. If the level read is below the value stored in THRESHOLD registers, then carrier presence is taken as positive. If the level read is above the value stored in THRESHOLD registers, then we assume there is no carrier presence.
  - ‘1’: Carrier Detection is detected automatically

11.3.9 CD_DECISION_TIME register

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
CD_DECISION_TIME	D7	D6	D5	D4	D3	D2	D1	D0

Name: CD_DECISION_TIME

Address: 0xFE49

Reset: “00100000”

- **D(7:0):** The number stored in this register is multiplied by (2^16/fclk) in order to know how long is the period (in seconds) between consecutive runs of the carrier detector decision algorithm when in automatic carrier detection mode (CD_ENABLE(0)=’1’).

Default value = 0x20 = 32 →

11.3.10 **CD_DECISION_ERROR register**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
CD_DECISION_ERROR	--	D6	D5	D4	D3	D2	D1	D0

Name: CD_DECISION_ERROR

Address: 0xFE4A

Reset: “00001111”

- **--:** Reserved bit
- **D(6:0):** This register sets the number of errors allowed before taking a decision in the automatic carrier detection algorithm.

  It is used only in automatic mode (CD_ENABLE(0)=’1’).

11.3.11 **TXRX_CTL register**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
TXRX_CTL	--	--	--	--	--	VALUE	HARD	POL

Name: TXRX_CTL

Address: 0xFE4B

Reset: “00000000”

- **--:** Reserved bit
- **VALUE:** This bit is writable by the user, and will be the output in the EMIT0 pin when HARD=’0’.
- **HARD:** Selects the signal to be output by EMIT0.
  - ‘0’: EMIT0 outputs the bit value stored in VALUE field
  - ‘1’: EMIT0 outputs the internal signal TXRX, with the polarity designated by POL
- **POL:** Polarity of EMIT0 when TXRX internal signal is selected as output (HARD=’1’)
  - ‘0’: ‘0’ is output in transmission, ‘1’ is output in reception
  - ‘1’: 1’ is output in transmission, ‘0’ is output in reception

11.3.12 **Ri registers**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	B it 0
R1	D7	D6	D5	D4	D3	D2	D1	D0
R2	D7	D6	D5	D4	D3	D2	D1	D0
R3	D7	D6	D5	D4	D3	D2	D1	D0
R4	D7	D6	D5	D4	D3	D2	D1	D0
R5	D7	D6	D5	D4	D3	D2	D1	D0
R6	D7	D6	D5	D4	D3	D2	D1	D0
R7	D7	D6	D5	D4	D3	D2	D1	D0
R8	D7	D6	D5	D4	D3	D2	D1	D0

Name: R1 - R8

Address: 0xFE4C – 0xFE53

Reset: R1:0x60

R2:0x60

R3:0x60

R4:0x60

R5:0xFF

R6:0xFF

R7:0xFF

R8:0xFF

The value in these registers strongly depends on the external configuration. ATMEL provides values to be used according with the design recommended in the ATMEL Reference Design.

Please contact ATMEL if other external configurations are going to be used.

Recommended register value	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**
	0x00	0x00	0x00	0x00	0xFF	0xFF	0xFF	0xF8

12. Electrical Characteristics

12.1 Absolute Maximum Ratings

Permanent device damage may occur if Absolute Maximum Ratings are exceeded. Functional operation should be restricted to the conditions given in the Recommended Operating Conditions section. Exposure to the Absolute Maximum Conditions for extended periods may affect device reliability.
(VSS = 0 V)

Table 12-1. **ATPL00B Absolute Maximum Ratings**

Parameter	Symbol	Rating	Unit
Supply Voltage	VCC	-0.5 to 4.0	V
Input Voltage	VI	-0.5 to VCC+0.5(≤4.0V)	V
Output Voltage	VO	-0.5 to VCC+0.5(<4.0V)	V
Storage Temperature	TST	-55 to 125	ºC
Junction Temperature	TJ	-40 to 125	ºC
Output Current (*)1	IO	±10 (*)2	mA

Notes: (*)1. DC current that continuously flows for 10ms or more, or average DC current.
(*)2. Applies to all the pins except EMIT pins. EMIT pins should be only used according to circuit configurations recommended by Atmel.

ATTENTION Observe ESD Precautions

Precautions for handling electrostatic sensitive devices should be taken into account to avoid malfunction. Charged devices and circuit boards can discharge without detection.

12.2 Recommended Operating Conditions

Table 12-2. **ATPL00B Recommended Operating Conditions**

Parameter	Symbol	Rating			Unit
		Min.	Typ.	Max.	
Supply Voltage	VCC	3.00	3.30	3.60	V
	VDEO	3.00	3.30	3.60	
	VDA	3.00	3.30	3.60	
Junction Temperature	TJ	-40	25	125	ºC

12.3 DC Characteristics

Table 12-3. **ATPL00B DC Characteristics**

Parameter	Condition	Symbol	Rating			Unit
			Min.	Typ.	Max.	
Supply Voltage		VCC	3.00	3.30	3.60	V
H-level Input Voltage (3.3v CMOS)		VIH	2.0	-	VCC+0.3	
L-level Input Voltage (3.3v CMOS)		VIL	-0.3	-	0.8	
H-level Output Voltage	3.3v I/O IOH=-100µA	VOH	VCC-0.2	-	VCC	
L-level Output Voltage	3.3v I/O IOL=100µA	VOL	0	-	0.2	
H-level Output V-I Characteristics	3.3v I/O VCC=3.3±0.3	IOH	Refer to Annex1			mA
L-level Output V-I Characteristics	3.3v I/O VCC=3.3±0.3	IOL	Refer to Annex1			
Internal Pull-up Resistor(*)1	3.3v I/O	Rpu	10	33	80	kΩ
Internal Pull-down Resistor(*)1	3.3v I/O	Rpd	10	33	80	kΩ
Junction Temperature		TJ	-40	-	125	ºC

Notes: (VCC=3.3v ± 0.3v , VSS=0v , TJ=-40 to 125°C)

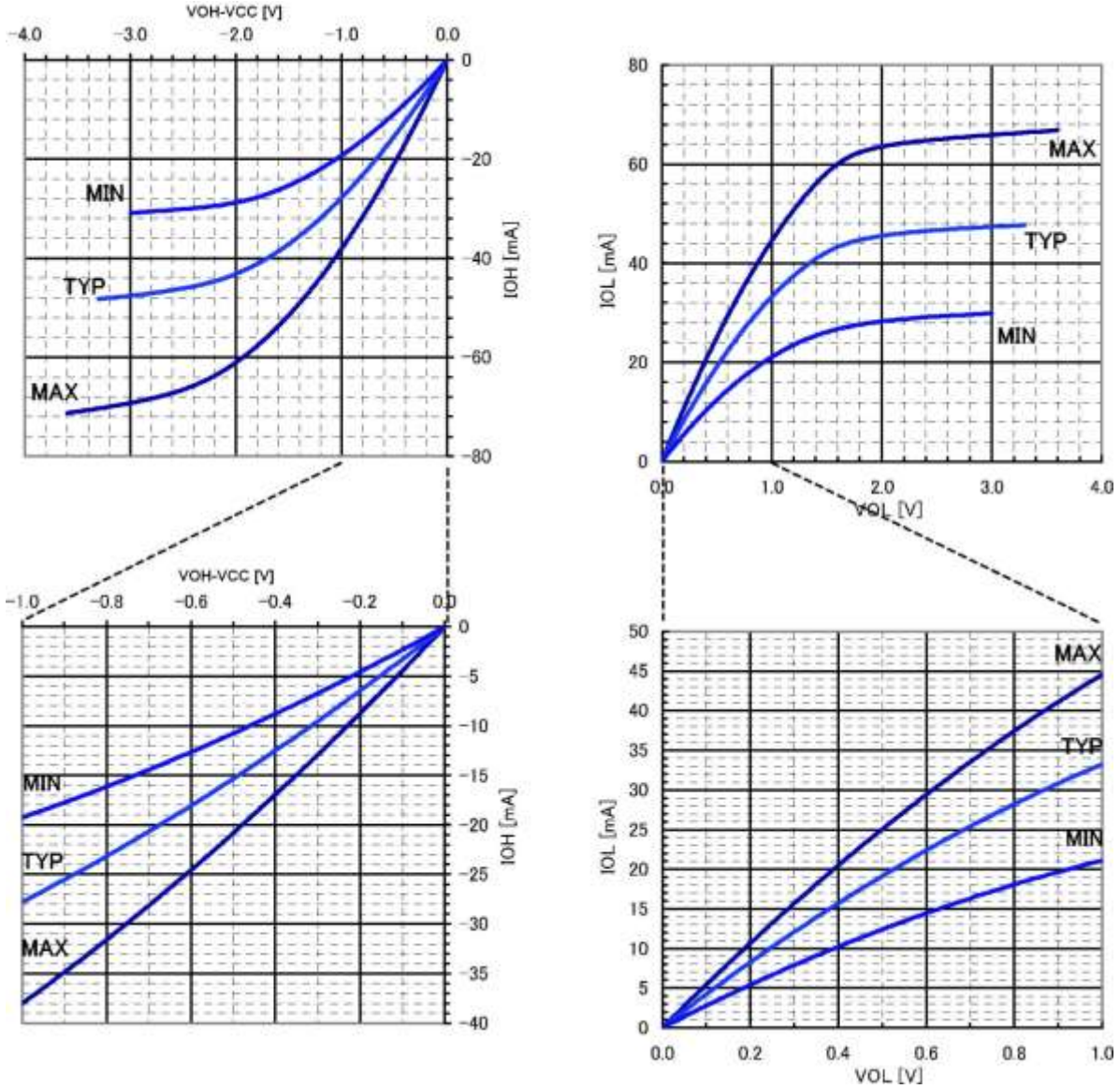
(*)1. Only applicable to pins with internal pulling. See related table. Table 2-1

12.3.2 V-I curves

V-I Characteristics 3.3 Vstandard CMOS IO L, M type
Pins marked in Table 2-1 - pinout table with Nominal Current I(mA)=±5

Condition:	MIN	Process=	Slow	Tj=	125°C	VCC=	3.0 V
	TYP	Process=Typical		Tj=	25°C	VCC=	3.3 V
	MAX	Process=Fast		Tj=	-40°C	VCC=	3.6 V

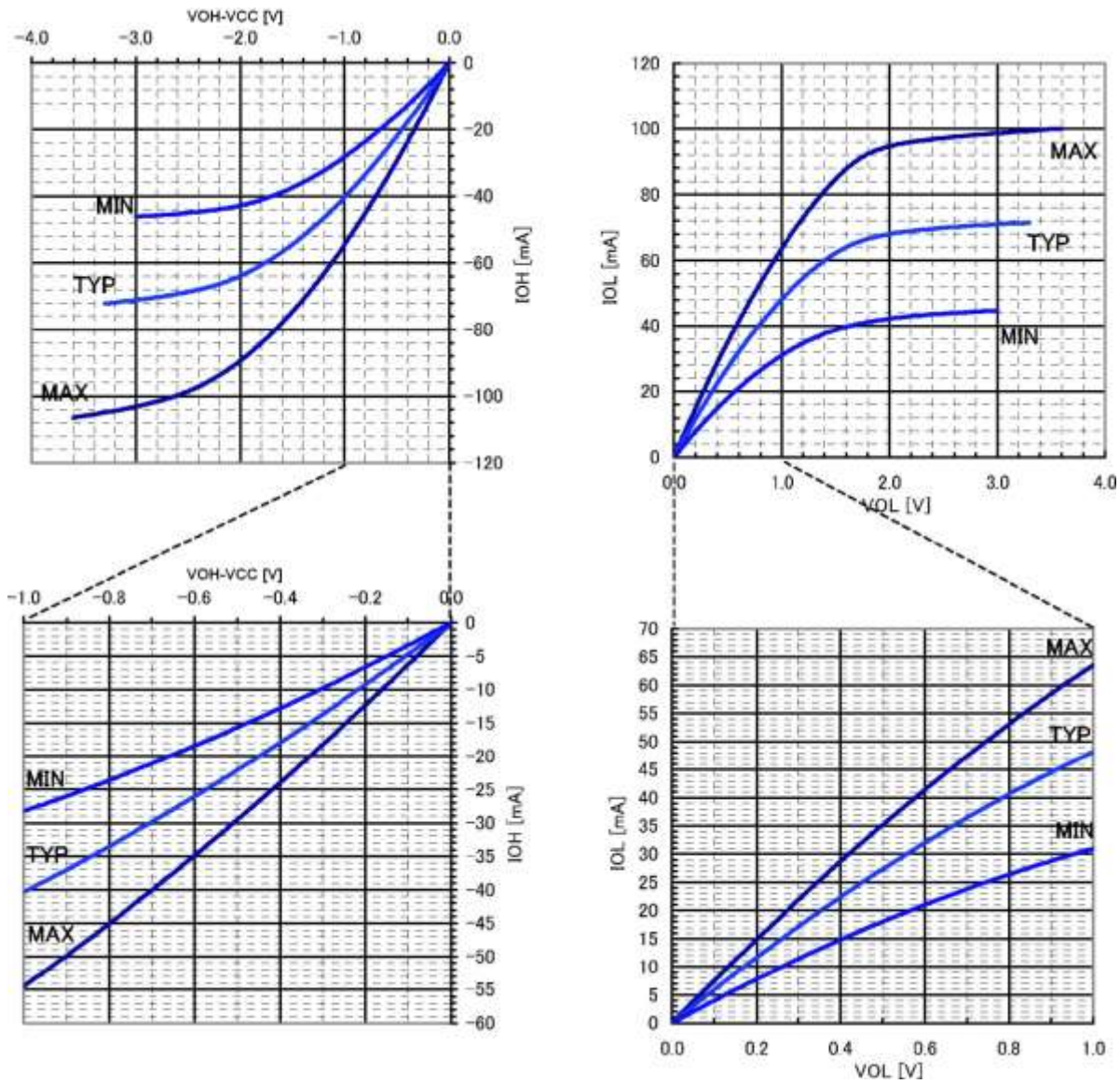
Figure 12-1. CMOS IO L and M type,V-I curves



V-I Characteristics 3.3 Vstandard CMOS IO H, V type
Pins marked in Table 2-1 - pinout table with Nominal Current I(mA)=±10

Condition:	MIN	Process=	Slow	Tj=	125°C	VCC=	3.0 V
	TYP	Process=Typical		Tj=	25°C	VCC=	3.3 V
	MAX	Process=Fast		Tj=	-40°C	VCC=	3.6 V

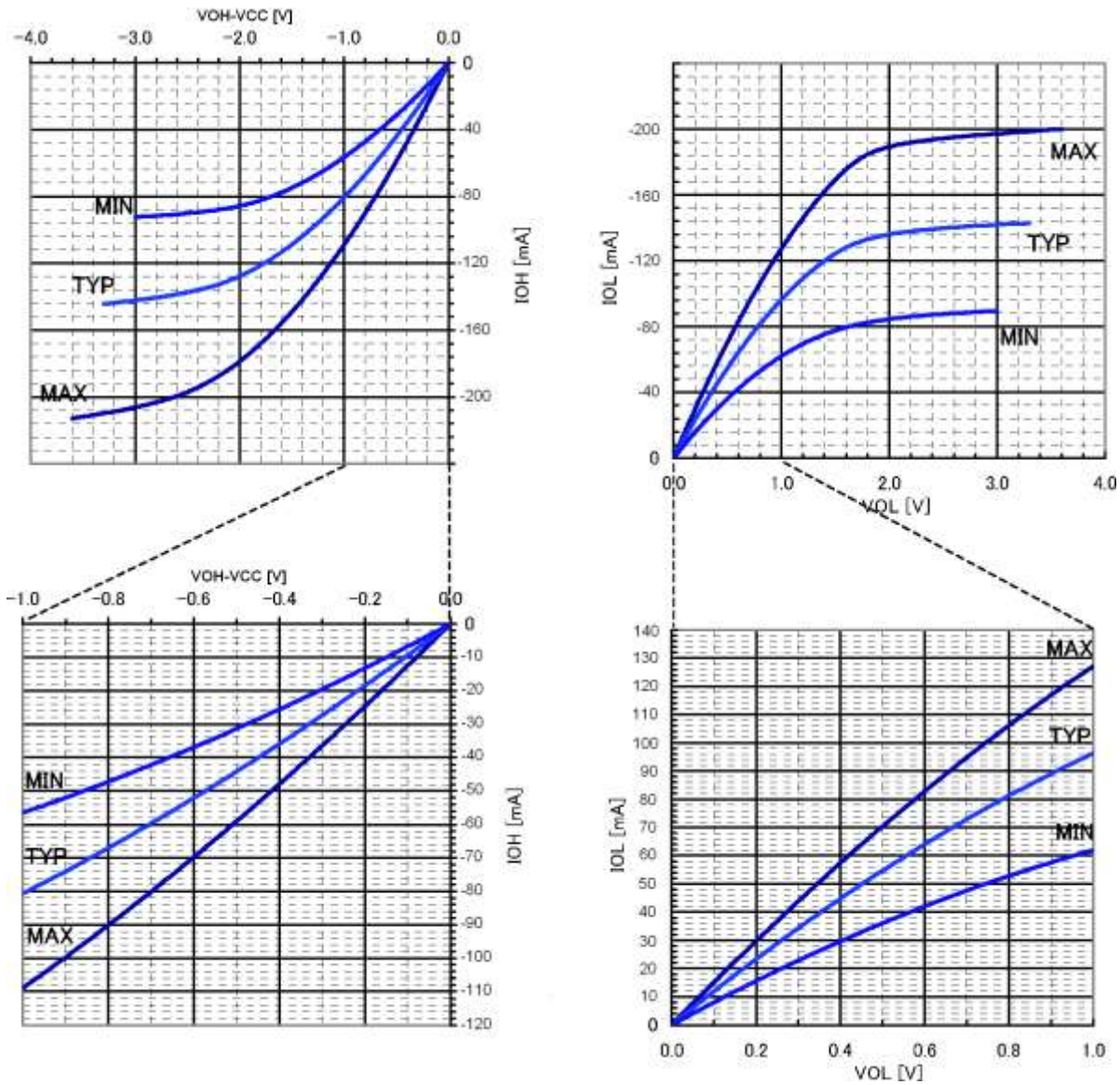
Figure 12-2. CMOS IO H and V type, V-I curves



V-I Characteristics 3.3 Vstandard CMOS IO H, V type
Pins marked in Table 2-1 - pinout table with Nominal Current I(mA)=±X

Condition:	MIN	Process=	Slow	Tj=	125°C	VCC=	3.0 V
	TYP	Process=Typical		Tj=	25°C	VCC=	3.3 V
	MAX	Process=Fast		Tj=	-40°C	VCC=	3.6 V

Figure 12-3. CMOS IO X type, V-I curves



12.4 Power Consumption

Table 12-4. **ATPL00B Power Consumption**

Parameter	Condition	Symbol	Rating			Unit
			Min.	Typ.	Max.	
Power Consumption in reception	TA=25°C, VCC=3.3v	P_{Rx25}	--	240	--	mW
Power Consumption in transmission	TA=25°C, VCC=3.3v	P_{Tx25}	--	455	--	mW
Power Consumption (worst case)	TA=85°C, VCC=3.6v	P_{85}	--	--	600*1,*2	mW

Notes: *1. Measured using recommended external configuration.

*2. RL=0R (Load Impedance=0Ω.)

12.5 Thermal Data

Table 12-5. **ATPL00B Thermal Data**

Parameter	Symbol	LQFP144	Unit
Thermal resistance junction-ambient steady state	$R_{Theta\text{-}ja}$	53 *(1) 37 *(2)	°C/W

Notes: 1. Mounted on 2-layer PCB.

2. Mounted on 4-layer PCB.

Theta-ja is calculated based on a standard JEDEC defined environment and is not reliable indicator of a device's thermal performance in a non-JEDEC environment. The customer should always perform their own calculations/simulations to ensure that their system's thermal performance is sufficient.

12.6 Oscillator

Figure 12-4. External Crystal configuration

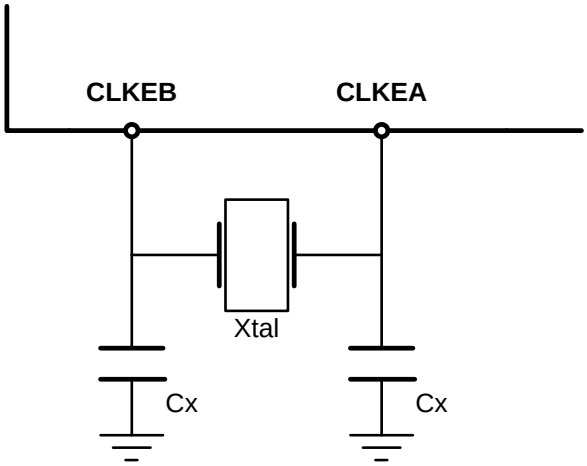


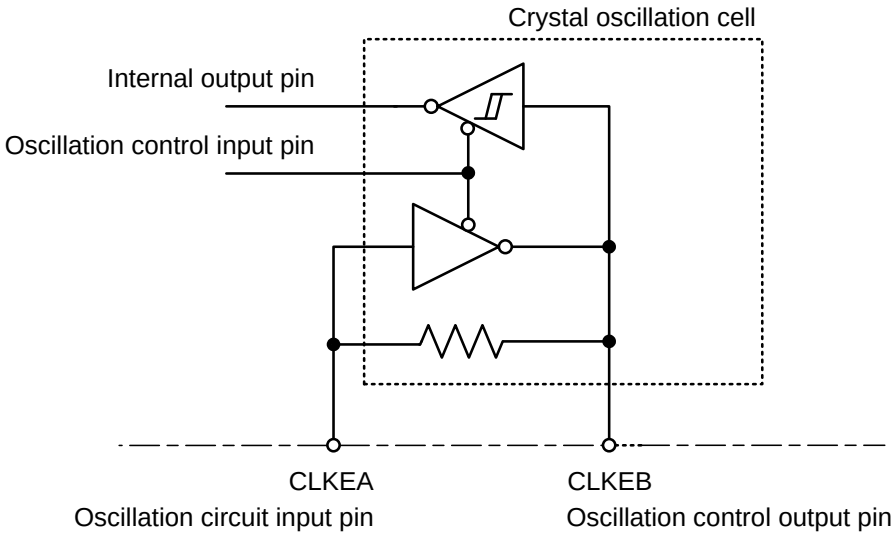
Table 12-6. **External oscillator parameters**

Parameter	Test Condition	Symbol	Rating			Unit
			Min.	Typ.	Max.	
Crystal Oscillator frequency	fundamental	Xtal	11.0592			MHz
External Oscillator Capacitance	See Figure 12-4	Cx	5	18	30	pF
H-level Input Voltage		XVIH	2	-	VCC+0.3	V
L-level Input Voltage		XVIL	-0.3	-	0.8	
External Oscillator Parallel Resistance		Rp	*not needed*			Ω
External Oscillator Series Resistance		Rs	*not needed*			

Notes:

1. The crystal should be located as close as possible to CLKEB and CLKEA pins.
2. Recommended value for Cx is 18pF. This value may depend on the specific crystal characteristics.
3. Crystal Stability/Tolerance/Ageing values must be selected according to standard PRIME requirements.

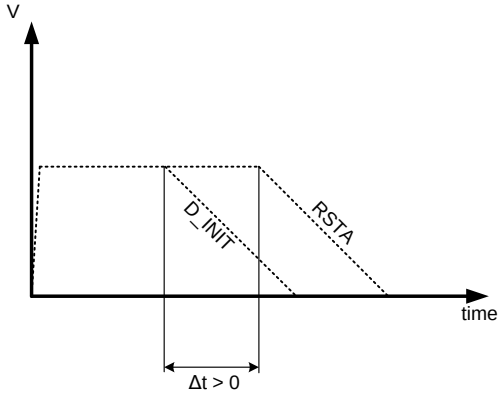
Figure 12-5. Internal Oscillator Cell



12.7 Power on

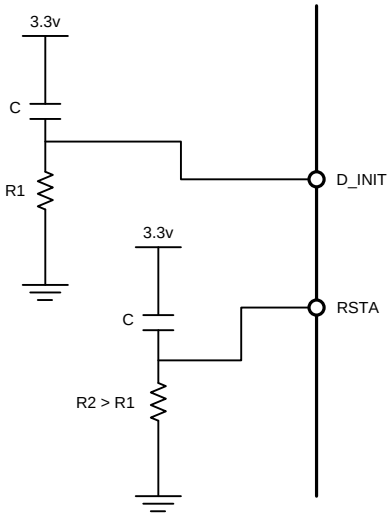
In power-on, D_INIT should be released before asynchronous reset signal RSTA in order to ensure proper system start up.

Figure 12-6. D_INIT & RSTA realease sequence during power-on



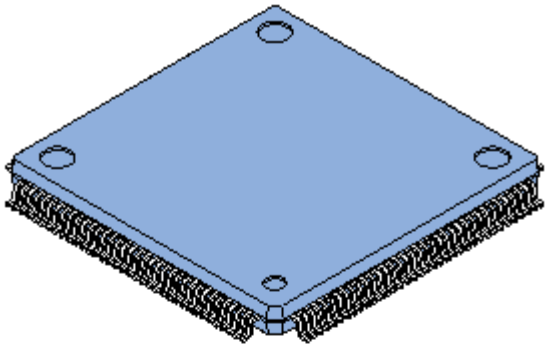
Not minimum time is required between both releases, Δt > 0, so a simple RC circuit is enough to satisfy this requirement.

Figure 12-7. Example circuit



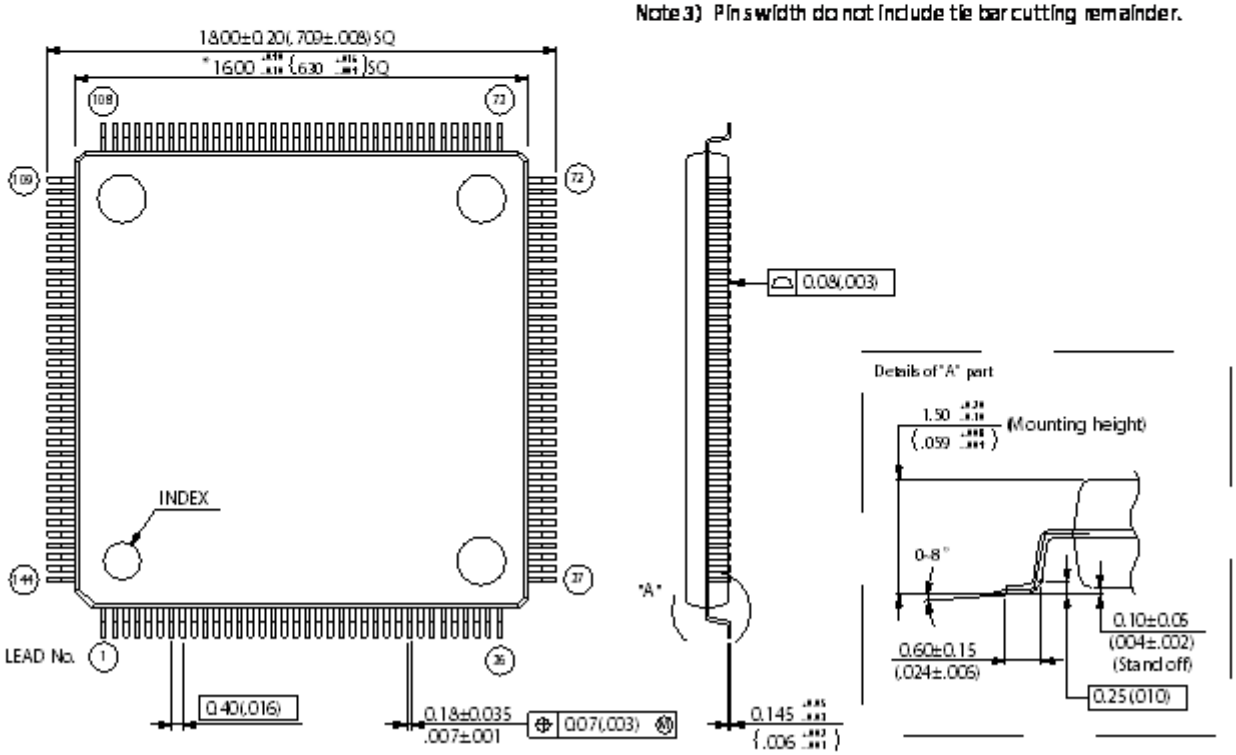
13. Mechanical Characteristics

Figure 13-1. 144-lead LQFP Package Mechanical Drawing



Lead pitch	0.40 mm
Package width · package length	16.0 · 16.0 mm
Lead shape	Gullwing
Sealing method	Plastic mold
Mounting height	1.70 mm MAX
Weight	0.88 g

Note 1) * : These dimensions include resin protrusion.
Resin protrusion is +0.25(.010)Max(each side).
Note 2) Pins width and pins thickness include plating thickness.
Note 3) Pins width do not include tie bar cutting remainder.



Dimensions in mm (inches).
Note: The values in parentheses are reference values.

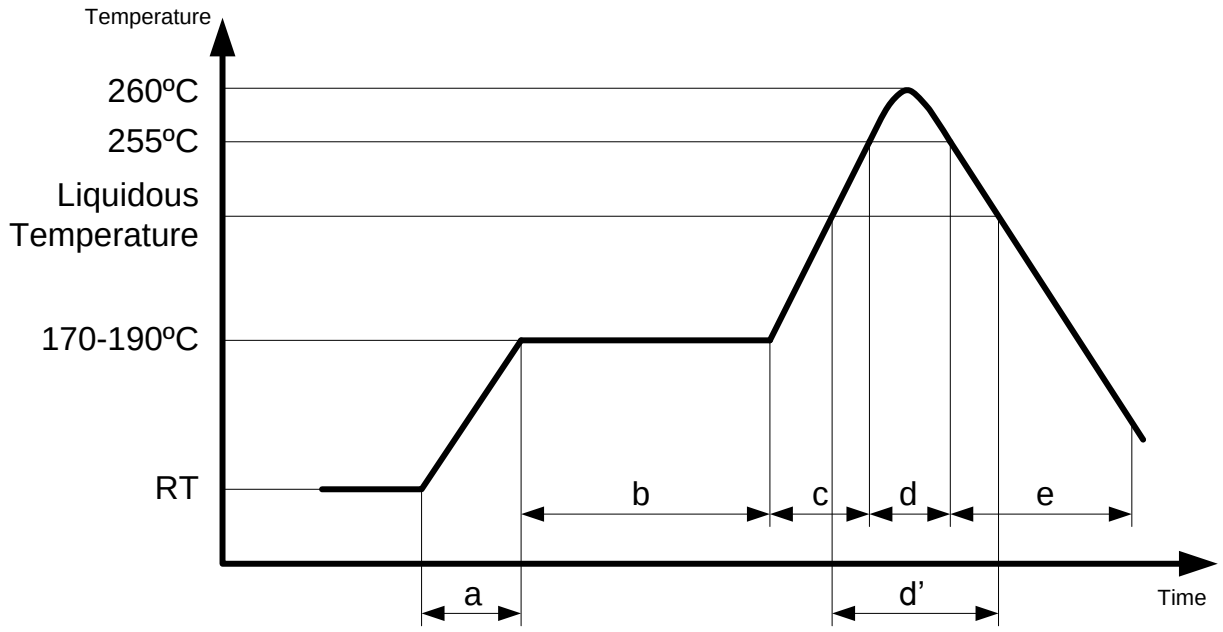
14. Recommended mounting conditions

14.1 Conditions of Standard Reflow

Table 14-1. **Conditions of standard Reflow**

Items	Contents	
Method	IR(Infrared Reflow)/Convection	
Times	2	
Floor Life	Before unpacking	Please use within 2 years after production
	From unpacking to second reflow	Within 8 days
	In case over period of floor life	Baking with 125ºC +/- 3ºC for 24hrs +2hrs/-0hrs is required. Then please use within 8 days. (please remember baking is up to 2 times)
Floor Life Condition	Between 5ºC and 30ºC and also below 70%RH required. (It is preferred lower humidity in the required temp range.)	

Figure 14-1. Temperature Profile



Note: H rank: 260ºC Max

a: Average ramp-up rate: 1ºC/s to 4ºC/s

b: Preheat & Soak: 170ºC to 190ºC, 60s to 180s

c: Average ramp-up rate: 1ºC/s to 4ºC

d: Peak temperature: 260ºC Max, up to 255ºC within 10s

d': Liquidous temperature: Up to 230ºC within 40s or
Up to 225ºC within 60s or
Up to 220ºC within 80s

e: Cooling: Natural cooling or forced cooling

14.2 Manual Soldering

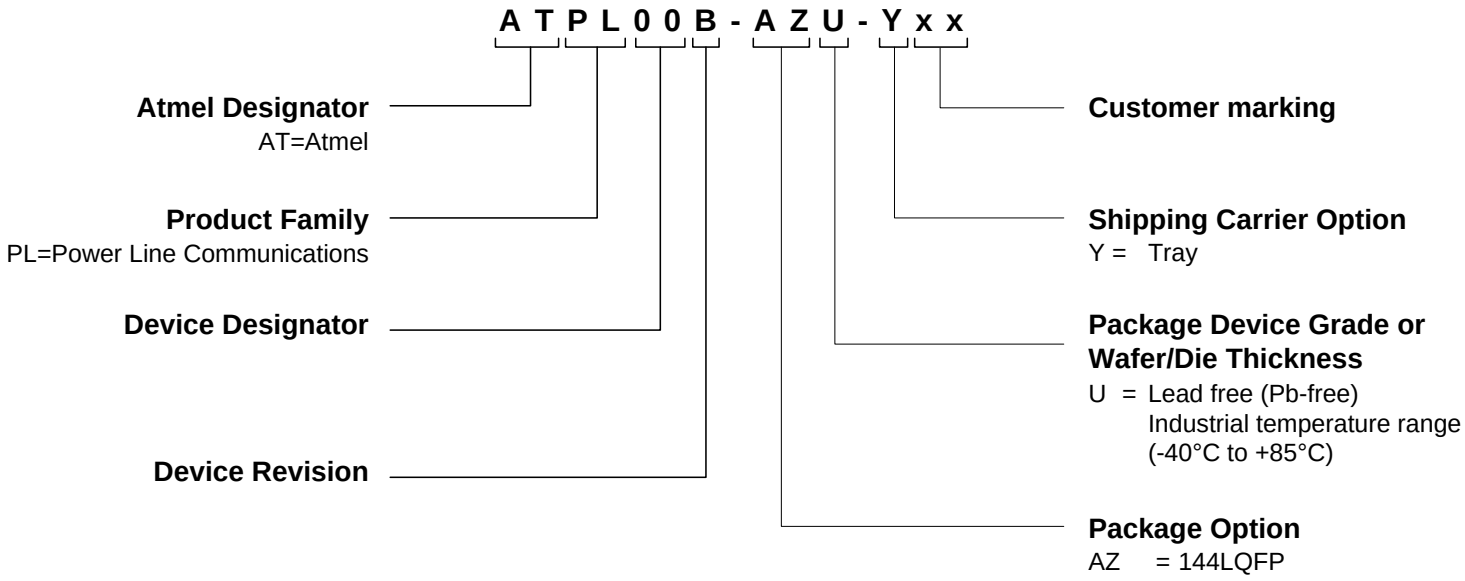
Table 14-2. **Conditions of Manual Soldering**

Items	Contents	
Floor life	Before unpacking	Please use within 2 years after production
	From unpacking to Manual Soldering	Within 2 years after production (No control required for moisture adsorption because it is partial heating)
Floor life condition	Between 5°C and 30°C and also below 70%RH required. (It is preferred lower humidity in the required temp range.)	
Solder Condition	Temperature of soldering iron: Max 400°C, Time: Within 5 seconds/pin *Be careful for touching package body with iron	

15. Ordering Information

Table 15-1. **Atmel ATPL00B Ordering Codes**

Atmel Ordering Code	Package	Package Type	Temperature Range
ATPL00B-AZU-Y	144 LQFP	Pb-Free	Industrial (-40ºC to 85º)



16. Revision History

Doc. Rev.	Date	Comments
A	09/27/2012	Initial Release

Enabling Unlimited Possibilities™

Atmel Corporation
1600 Technology Drive
San Jose, CA 95110
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan G.K.
16F Shin-Osaki Kangyo Building
1-6-4 Osaki
Shinagawa-ku, Tokyo 141-0032
JAPAN
Tel: (+81)(3) 6417-0300
Fax: (+81)(3) 6417-0370

© 2012 Atmel Corporation. All rights reserved. / Rev.: 43028A-ATPL - 9/12

Atmel®, Atmel logo and combinations thereof, and others are registered trademarks or trademarks of Atmel Corporation or its subsidiaries. Other terms and product names may be trademarks of others.

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN THE ATMEL TERMS AND CONDITIONS OF SALES LOCATED ON THE ATMEL WEBSITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS AND PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and products descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.