Y51 Microcontroller

Technical Manual

Disclaimer

Systemyde International Corporation reserves the right to make changes at any time, without notice, to improve design or performance and provide the best product possible. Systemyde International Corporation makes no warrant for the use of its products and assumes no responsibility for any errors which may appear in this document nor does it make any commitment to update the information contained herein.

Systemyde International Corporation products are not authorized for use in life support devices or systems unless a specific written agreement pertaining to such use is executed between the manufacturer and the President of Systemyde International Corporation. Nothing contained herein shall be construed as a recommendation to use any product in violation of existing patents, copyrights or other rights of third parties. No license is granted by implication or otherwise under any patent, patent rights or other rights, of Systemyde International Corporation. All trademarks are trademarks of their respective companies.

Every effort has been made to ensure the accuracy of the information contain herein. If you find errors or inconsistencies please bring them to our attention. In all cases, however, the Verilog HDL source code for the Y51 design defines “proper operation”.

Notice:

“Intel” and “8051” are registered trademarks of Intel, Inc. All uses of these terms in this document are to be construed as adjectives, whether or not the noun “microprocessor”, “microcontroller”, “CPU” or “device” are actually present.

Table of Contents

Revison History

Date	Changes	Page(s)
10/20/2012	Preliminary issue	
06/28/2013	Added details for peripherals	
07/18/2013	corrected JBC operation details; typos and spelling	
08/16/2013	TCON register description	94
08/22/2013	Timer operation	

Introduction

This publication documents the operation of the Y51 microcontroller. This CPU design is supplied in Verilog HDL and can be implemented in any technology supported by a logic synthesis tool that accepts Verilog HDL. Included in the design package is a test bench that exercises all instructions, flag settings, and representative data patterns. The test patterns should achieve at least 95% fault coverage.

The Y51 CPU was designed in a clean-room environment and is compatible with the Intel 8051 microcontroller. Only publicly available documentation was used to create this design so there may be minor differences where the public documentation is misleading or lacking. The instruction execution times are not identical between the two designs. The Y51 CPU operates with a consistent two-clock-cycle machine cycle, while the 8051 microcontroller uses a machine cycle with six clock cycles.

This document should always be used as the final word on the operation of the Y51 CPU, but it is useful to refer to the Intel documentation if the description given here is too cryptic. The 8051 architecture is over thirty years old, so it is assumed that it is already at least somewhat familiar to the reader.

The Y51 CPU is accompanied by full design documentation, in the form of a large spreadsheet, which describes nearly every facet of the internal operation of the processor. This provides knowledgeable users the opportunity to customize the design for unique application requirements.

Features

* Fully functional synthesizable Verilog HDL version of the 8051/8052 CPU

* Vendor and technology independent

* Software compatible with several industry-standard processors

* Static, fully synchronous design uses no 3-state buses

* Uniform 2 clock-cycle machine cycle

* All memory interfaces match common synchronous FPGA and ASIC memory timing

* Program memory and External RAM memory share address and data buses

* Separate strobes for Program Memory and External RAM memory

* Separate Special Function Register (SFR) bus with dedicated strobes

* Internal RAM uses FPGA memory macros

* Four Parallel Ports

* Two Timer/Counters

* One serial channel

* Two additional internal interrupts for additional peripherals

* Full design documentation included

* Verilog simulation and test suite included

Interface Description

This CPU design makes no attempt to match the signals or timing present on the 8051 microprocessor. Rather, all of the bus interfaces and signals are optimized for use in either an ASIC or an FPGA. The interface signals for the design are detailed below.

CLOCK and RESET

The design uses a single clock and single reset. No clock gating is employed in the design.

clk (input, active-High) The rising edge of the Master Clock samples all inputs except for the asynchronous reset and all outputs normally change in response to the rising edge of the Master Clock.

resetb (input, active-Low) The Master Reset signal is used to initialize all state flip-flops, and user registers consistent with the original 8051 design. This is an asynchronous signal, but the trailing edge should be synchronized with the rising edge of the Master Clock.

PROGRAM MEMORY and EXTERNAL RAM

The interface for Program Memory and External RAM consists of a single 16-bit address bus and separate buses for read data and write data. Program Memory and External RAM accesses use separate strobes. The cycle time for this these buses is two clock cycles.

mem_addr (output, 16-bit bus) The Memory Address bus carries the address all Program Memory read transactions, as well as External RAM read and write transactions.

mem_rdata (input, 8-bit bus) The Memory Read Data bus is sampled during Program Memory and External RAM read transactions.

mem_rd (output, active-High) The Memory Read signal is one clock cycle wide and identifies the data transfer clock cycle for Program Memory read transactions. There is no corresponding Memory Write signal, because the 8051 architecture does not support writing to Program Memory.

mem_wdata (output, 8-bit bus) The Memory Write Data bus carries the output data for External RAM write transactions. To conserve power, this bus only changes state as required for a memory write.

ph (output, active-High) The Bus Phase signal is High during the first clock of a machine cycle and Low during the second clock of a machine cycle.

xram_rd (output, active-High) The External RAM Read signal is one clock cycle wide and identifies the data transfer clock cycle for External RAM read transactions.

xram_wr (output, active-High) The External RAM Write signal is one clock cycle wide and identifies the data transfer clock cycle for External RAM write transactions.

SPECIAL FUNCTION REGISTERS

Access to external Special Function Registers is via the SFR bus. The SFR bus operates with a one-clock cycle time, and simultaneous reads and writes are not possible. Internal Special Function Registers do not use this bus, but are directly connected internally in the design.

sfr_addr (output, 7-bit bus) The Special Function Register Address bus carries the address for Special Function Register transactions. This bus only valid during Special Function Register transfers.

sfr_rd (output, active-High) The Special Function Register Read Enable signal identifies data transfer clock cycles for Special Function Register read transactions.

sfr_rdata (input, 8-bit bus) The Special Function Register Read Data bus is sampled by the clock when the **sfr_rd** signal is active and an external Special Function Register is addressed.

sfr_wdata (output, 8-bit bus) The Special Function Register Write Data bus carries the output data for Special Function Register write transactions. This bus is valid only during Special Function write operations.

sfr_wr (output, active-High) The Special Function Register Write Enable signal identifies data transfer clock cycles for Special Function Register write transactions.

INTERRUPTS

The design supports two external interrupt requests, and provides an output to signal that an interrupt acknowledge is in progress. The two “internal” interrupt requests are intended for use with on-chip peripherals beyond those native to the design.

ext0_int, ext1_int (input, active-High) The standard External Interrupt Request signals can be independently programmed to be either level-sensitive or edge-triggered. In the case of a level triggering the interrupt request must be de-asserted before the end of the interrupt service routine. Edge-triggered interrupts are sampled by the Master Clock to recognize a rising edge. This information is latched until the interrupt is acknowledged, at which point the hardware will automatically clear the latched status.

int0_int, int1_int (input, active-High) The new Internal Interrupt Request signals are always level-sensitive. These inputs are not synchronized, because it is assumed that they will be generated by on-chip peripheral devices.

intack (output, active-High) The Interrupt Acknowledge signal is active for two clock cycles to indicate that an interrupt acknowledge sequence is in progress.

PARALLEL PORTS

The parallel port functionality is implemented as separate inputs and outputs. There is no multiplexing with address or data information or other peripheral functionality, to make the design as general-purpose as possible.

p0_in, p1_in, p2_in, p3_in (input, 8-bit bus) The state of the Port inputs is returned during reads of the port data registers unless the read is part of a read-modify-write instruction. In this case the port output data register will be returned to allow for proper modify operation.

p0_out, p1_out, p2_out, p3_out (output, 8-bit bus) The Port outputs reflect the contents of the port output data registers.

TIMER/COUNTERS

The design includes the two timer/counters found in the 8052. To ensure backwards compatibility, these timers are clocked at 1/12 the rate of the Master Clock.

t0_cnt, t1_cnt (input, active-High) The Timer x Count signals are used in the counter mode of operation to cause the counter to increment. After synchronization,

these inputs are sampled at the timer clock rate to detect a falling edge. The falling edge cause the counter to increment.

t0_gate, t1_gate (input, active-High) The Timer x Gate signals are used to enable the count signal, in either timer or counter mode. This function is enabled or disabled under program control.

t0_out, t1_out (output, active-High) The Timer x Output signals are active for one clock cycle (of the Master Clock) when the counter/timer overflows or is reloaded.

SERIAL CHANNEL

The design includes the basic serial channel found in the 8051. Backwards-compatible enhancements provide more error reporting and improved features.

rxd_in (input, active-High) The Receive Data Input signal is the data input for the serial channel.

syn_dir (output) The Synchronous Direction signal indicates the transfer direction for the serial data in the synchronous mode. Low indicates data reception and High indicates data transmission.

txc_out (output, active-High) The Transmit Clock Output signal outputs the data clock in the synchronous mode. This signal is inactive (Low) when the serial channel is in a UART mode.

txd_out (output, active-High) The Transmit Data Output signal is the data output for the serial channel.

UNCOMMITTED OUTPUTS

The design includes two uncommitted 8-bit Special Function Registers, called PCON (Power Control) and ECOM (Extended Control), which can be used in a system for hardware control.

econ_reg (output, 8-bit bus) The Extended Control register is intended for hardware control of logic external to the CPU.

pcon_reg (output, 8-bit bus) The Power Control register is intended for hardware control of power-control logic external to the CPU.

External Timing

This CPU design uses a uniform two-clock-cycle machine cycle. This consistent timing simplifies the design of logic external to the CPU makes it easier to track the state of the CPU.

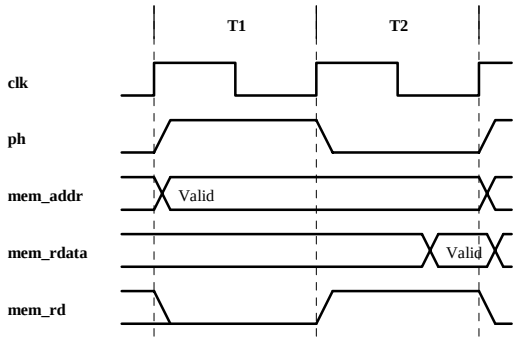
The program memory interface timing and signals are designed to make it easy to interface to standard ASIC and FPGA memories. This interface uses separate read and write strobes.

The SFR interface is similar to the AMBA Peripheral Bus (APB), except that it uses separate read and write strobes. The only timing difference relative to the APB is the setup time for the write data. In the APB the write data is setup one clock before the strobe; in this interface the write data changes coincident with the leading edge of the strobe. In most cases this will not be a problem.

In the diagrams below only the relevant signals are shown for each transaction. All other signals are either inactive or hold the previous value.

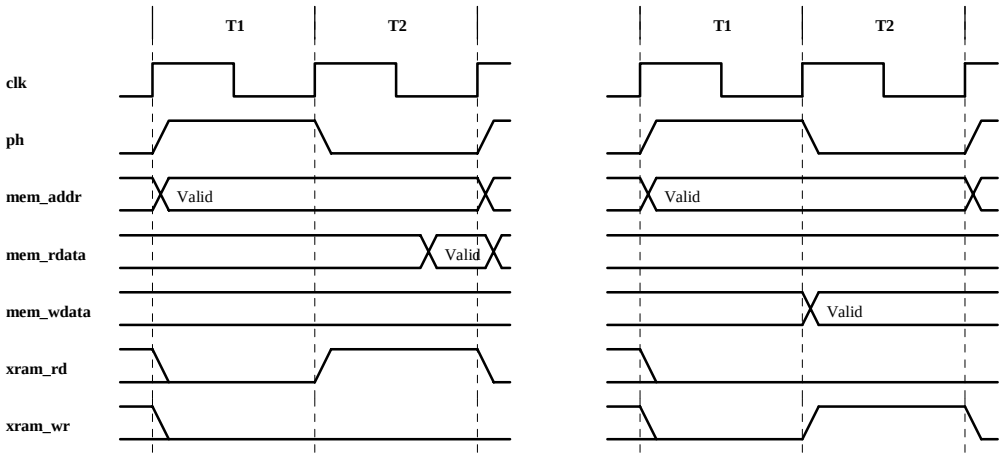
Program Memory Read

The figure below shows a Program Memory read transaction. Program Memory is read-only.



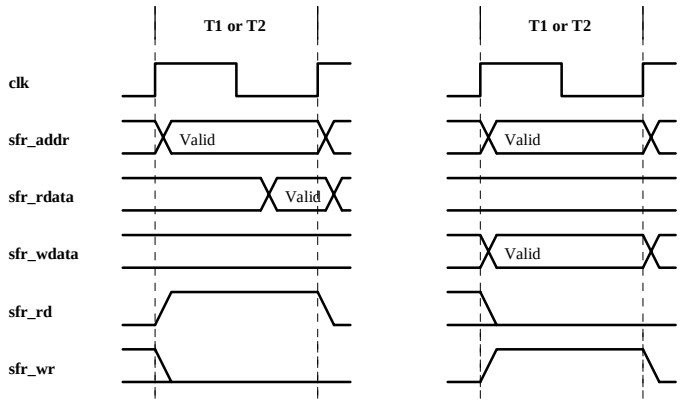
External RAM Read and Write

The figures below show read and write transactions with External RAM.



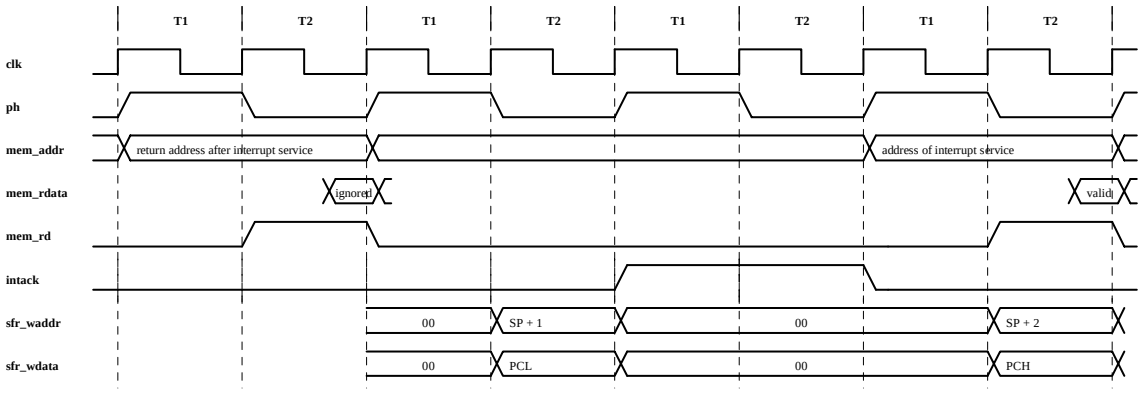
SFR Read and Write

The figures below show read and write transactions on the SFR interface. Unlike the Program Memory and External RAM interfaces, the SFR interface uses one clock-cycle timing. The **sfr_addr** and **sfr_wdata** buses are driven with all zeros when not in use, but since they are also used to carry information to the internal Special Function Registers as well as the Internal RAM, these buses will carry non-zero information at other times. External Special Function Registers must use the **sfr_rd** and **sfr_wr** strobes to read and write information.



Interrupt Acknowledge

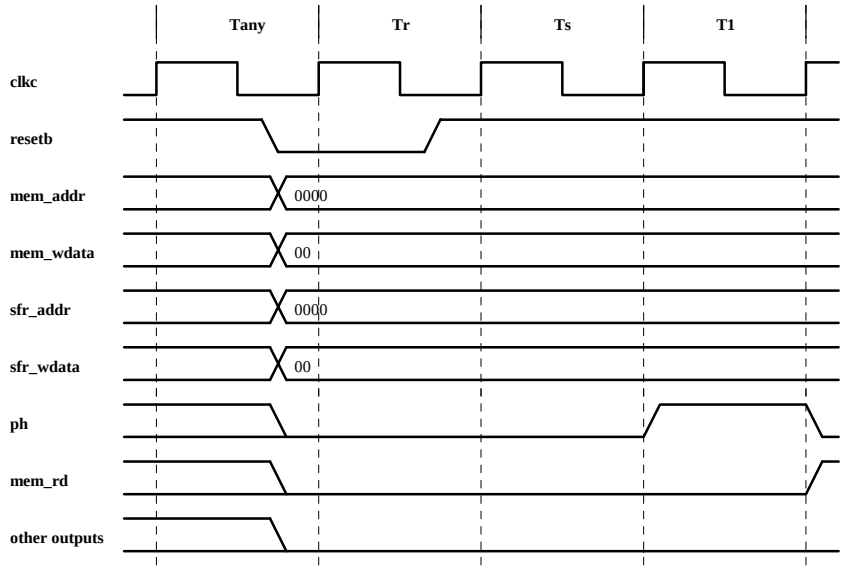
The figure below shows the interrupt acknowledge transaction, including the preceding aborted instruction fetch and dummy cycle, and subsequent instruction fetch for the service routine. The **sfr_addr** and **sfr_wdata** buses are also shown to indicate where the internal RAM is written.



Reset

The Reset state is entered immediately when the **resetb** signal goes Low, independent of the current state, and this state continues until the first rising edge of **clk** after the **resetb** signal is de-asserted. At this rising edge there is a one clock cycle transient state to set up the internal pipeline controls, and on the next clock the processor begins fetching the first instruction from address 0x0000.

The minimum width of the **resetb** signal is set by the flip-flops used in the design. The setup time for the **resetb** signal to the rising edge of the **clk** signal is likewise determined by the flip-flops used in the design.



Instruction Set

This chapter presents the assembly language syntax, addressing modes, flag settings, binary encoding, and execution time for the Y51 instruction set. The entire instruction set is presented in alphabetical order.

The assembly language syntax is identical to that used by the original Intel assembler. Different assembler programs may or may not use identical syntax. The syntax is presented generically at the beginning of each instruction, with the details presented for each addressing mode later in each entry.

The operation of each instruction is specified in a format similar to Verilog HDL for minimum ambiguity, but no descriptive text or examples are included.

The effect of the instruction on each flag is listed, with a brief description. The flags are organized as shown below in the PSW (Processor Status Word) register:

CY	AC	F0	RS1	RS0	OV	F1	P

These flags have the following meanings:

Flag	Meaning
CY	Carry (arithmetic carry, shift linkage bit, bit operation result).
AC	Auxiliary Carry (carry out of the lower nibble, used for BCD math).
F1, F0	User Flags 1 and 0.
RS1, RS0	Register Select control.
OV	Overflow (arithmetic overflow).
P	Parity.

Fields in the instruction are listed using shortcuts for common fields. These shortcuts should be self-explanatory in most cases, but will be detailed here for completeness.

The most common field in the instruction specifies a CPU register, employing the following encoding:

rrr	Register selected
000	R0
001	R1
010	R2
011	R3
100	R4
101	R5
110	R6
111	R7

Indirect registers are similarly encoded:

i	Indirect Register selected
0	R0
1	R1

The execution time for instructions is always a multiple of two clocks.

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	
F8	IP1								FF
F0	B								F7
E8									EF
E0	ACC								E7
D8									DF
D0	PSW								D7
C8									CF
C0									C7
B8	IP0								BF
B0	P3								B7
A8	IE								AF
A0	P2								A7
98	SCON	SBUF	BRLL	BRLH	BRCON	ESCON	SADDR	SADEN	9F
90	P1								97
88	TCON	TMOD	TL0	TL1	TH0	TH1	RCON	ECON	8F
80	P0	SP	DPL	DPH	DP1L	DP1H	DPS	PCON	87
78									7F
70									77
68									6F
60									67
58									5F
50									57
48									4F
40									47
38									3F
30									37
28									2F
20									27
18	R0	R1	R2	R3	R4	R5	R6	R7	1F
10	R0	R1	R2	R3	R4	R5	R6	R7	17
08	R0	R1	R2	R3	R4	R5	R6	R7	0F
00	R0	R1	R2	R3	R4	R5	R6	R7	07

CPU register set 0
CPU register set 1
CPU register set 2
CPU register set 3
Bit-addressable registers
External Special Function Registers

Notes:

1. Addresses 0x00-0x7F can be accessed using either a direct address or an indirect address.

2. Indirect addressing with addresses 0x80-0xFF accesses RAM.

3. Direct addressing with addresses 0x80-0xFF access the Special Function Registers.

4. RAM addresses 0x80-0xFF support multiple banks, depending on the technology used for the implementation.

5. Bit-addressed operations access addresses 0x20-0x2F in RAM and addresses 0x80, 0x88, 0x90... 0xE8, 0xF0 and 0xF8 in the Special Function Registers.

ACALL

Absolute Call

ACALL addr11

Operation:
(SP+1) <= PCL
(SP+2) <= PCH
PC[10:0] <= addr11
SP <= SP+2

Flags:
C: Unaffected
AC: Unaffected
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	ACALL addr11	aaa10001 addr[7:0]	4

Notes:

1. Bits 7:5 in the first opcode contain addr11[11:8].

2. The PC value used for PC[15:12] is the address of the next instruction.

ADD

Add

ADD A, src — src: R, DA, IR, IM

Operation: A <= A + src

Flags:
C: Set if arithmetic carry out of bit 7; cleared otherwise.
AC: Set if arithmetic carry out of bit 3; cleared otherwise.
OV: Set if arithmetic overflow; cleared otherwise.

Addressing Modes	Assembly Syntax	Encoding	Clocks
R:	ADD A, Rn	00101rrr	2
DA:	ADD A, direct	00100101 direct address	4
IR:	ADD A, @Ri	0010011i	4
IM:	ADD A, #data	00100100 immediate data	4

ADDC

Add with Carry

ADDC A, src src: R, DA, IR, IM

Operation: A <= A + src + C

Flags:
C: Set if arithmetic carry out of bit 7; cleared otherwise.
AC: Set if arithmetic carry out of bit 3; cleared otherwise.
OV: Set if arithmetic overflow; cleared otherwise.

Addressing Modes	Assembly Syntax	Encoding	Clocks
R:	ADDC A, Rn	00111rrr	2
DA:	ADDC A, direct	00110101 direct address	4
IR:	ADDC A, @Ri	0011011i	4
IM:	ADDC A, #data	00110100 immediate data	4

AJMP

Absolute Jump

AJMP addr11

Operation: PC[11:0] <= addr11

Flags:
C: Unaffected
AC: Unaffected
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	AJMP addr11	aaa00001 addr[7:0]	4

Notes:

1. Bits 7:5 in the first opcode contain addr11[11:8].

2. The PC value used for PC[15:12] is the address of the next instruction.

ANL

Logical AND

ANL A, src — src: R, DA, IR, IM

Operation: A <= A & src

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
R:	ANL A, Rn	01011rrr	2
DA:	ANL A, direct	01010101 direct address	4
IR:	ANL A, @Ri	0101011i	4
IM:	ANL A, #data	01010100 immediate data	4

ANL

Logical AND with Memory

ANL direct, src src: A, IM

Operation: direct <= direct & src

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
A	ANL direct, A	01010010 direct address	4
IM	ANL direct, #data	01010011 direct address immediate data	6

Notes:

1. This is a read-modify-write instruction, so when reading a Port register the data returned by the Port register is the contents of the output register rather than the input register.

ANL

Logical AND with Carry

ANL C, src src: bit, /bit

Operation: C <= C & src, where src can be either a bit or the complement of a bit

Flags: **C:** Set/cleared with the result of the operation.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
Bit:	ANL C, bit	10000010 bit address	4
Bit:	ANL C, /bit	10110000 bit address	4

CJNE

Compare and Jump if Not Equal

CJNE dst, src, offset

src: DA, IM
dst: A, DA

Operation:

```
if (dst != src) then begin
    PC <= PC + offset
    if (dst < src) then C <= 1 else C <= 0
    end
```

Flags: **C:** Set if dst < src; cleared otherwise.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
DA:	CJNE A, src, offset	10110101 direct address address offset	8
IM:	CJNE A, #data, offset	10110100 immediate data address offset	8
IM:	CJNE Rn, #data, offset	10111rrr immediate data address offset	8
IM:	CJNE @Ri, #data, offset	1011011i immediate data address offset	8

Notes:

1. The PC value used in the address calculation is the address of the next instruction.

2. For the comparison, the src and dst are treated as unsigned numbers.

CLR

Clear

CLR A

Operation: A <= 8’h00

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	CLR A	11100100	2

CLR

Clear Bit

CLR dst dst: C, bit

Operation: dst <= 1'b0

Flags: **C:** Cleared if dst is C; unaffected otherwise.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
C:	CLR C	11000011	2
Bit:	CLR bit	11000010 bit address	4

Notes:

1. This is a read-modify-write instruction, so when reading a Port register the data returned by the Port register is the contents of the output register rather than the input register.

CPL

Complement

CPL A

Operation: A <= ~A

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	CPL A	11110100	2

CPL

Complement Bit

CPL dst dst: C, bit

Operation: dst <= ~dst

Flags: **C:** Complemented if dst is C; unaffected otherwise
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
C:	CPL C	10110011	2
Bit:	CPL bit	10110010 bit address	4

Notes:

1. This is a read-modify-write instruction, so when reading a Port register the data returned by the Port register is the contents of the output register rather than the input register.

DA

Decimal Adjust

DA A

Operation: A <= Decimal Adjust A

Flags: **C:** Set if result is negative; cleared otherwise.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	DA A	11010100	2

Notes:

C before DA	A[7:4] before DA	AC before DA	A[3:0] before DA	Number added to A	C after DA
0	0-9	0	0-9	00	0
0	0-8	0	A-F	06	0
0	0-9	1	0-3	06	0
0	A-F	0	0-9	60	1
0	9-F	0	A-F	66	1
0	A-F	1	0-3	66	1
1	0-2	0	0-9	60	1
1	0-2	0	A-F	66	1
1	0-3	1	0-3	66	1

DEC

Decrement

DEC dst — dst: A, R, DA, IR

Operation: dst <= dst - 1

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
A:	DEC A	00010100	2
R:	DEC Rn	00011rrr	2
DA:	DEC direct	00010101 direct address	4
IR:	DEC @Ri	0001011i	4

Notes:

1. This is a read-modify-write instruction, so when reading a Port register the data returned by the Port register is the contents of the output register rather than the input register.

DIV

Divide

DIV AB

Operation: A <= A / B
B <= rem (A / B)

Flags: **C:** Cleared.
AC: Unaffected.
OV: Set if B = 0 before operation; cleared otherwise.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	DIV AB	10000100	6

Notes:

1. Both A and B are treated as unsigned numbers for the division.

DJNZ

Decrement, Jump if Non-Zero

DJNZ dst, offset dst: R, DA

Operation: dst <= dst - 1
if (dst != 0) then PC <= PC + offset

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
R:	DJNZ Rn, offset	11011rrr address offset	6
DA:	DJNZ direct, offset	11010101 direct address address offset	8

Notes:

1. The PC value used in the address calculation is the address of the next instruction.

2. This is a read-modify-write instruction, so when reading a Port register the data returned by the Port register is the contents of the output register rather than the input register.

INC

Increment

INC dst — dst: A, R, DA, IR

Operation: dst <= dst + 1

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
A:	INC A	00000100	2
R:	INC Rn	00001rrr	2
DA:	INC direct	00000101 direct address	4
IR:	INC @Ri	0000011i	4

Notes:

1. This is a read-modify-write instruction, so when reading a Port register the data returned by the Port register is the contents of the output register rather than the input register.

INC

Increment DPTR

INC DPTR

Operation: DPTR <= DPTR + 1

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	INC DPTR	10100011	2

JB

Jump if Bit Set

JB bit, offset

Operation: if (bit) then PC <= PC + offset

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	JB bit, offset	00100000 bit address address offset	8

Notes:

1. The PC value used in the address calculation is the address of the next instruction.

JBC

Jump if Bit Set and Clear Bit

JBC bit, offset

Operation: if (bit) then PC <= PC + offset
bit <= 1'b0

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	JBC bit, offset	00010000 bit address address offset	8

Notes:

1. The PC value used in the address calculation is the address of the next instruction.

2. This is a read-modify-write instruction, so when reading a Port register the data returned by the Port register is the contents of the output register rather than the input register.

JC

Jump if Carry Set

JC bit, offset

Operation: if (C) then PC <= PC + offset

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	JC bit, offset	01000000 address offset	6

Notes:

1. The PC value used in the address calculation is the address of the next instruction.

JMP

Jump Indirect

JMP @A+DPTR

Operation: PC <= A + DPTR

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	JMP @A+DPTR	01110011	4

Notes:

1. A is treated as an unsigned number for the addition.

JNB

Jump if Bit Clear

JNB bit, offset

Operation: if (!bit) then PC <= PC + offset

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	JNB bit, offset	00110000 bit address address offset	8

Notes:

1. The PC value used in the address calculation is the address of the next instruction.

JNC

Jump if Carry Clear

JNC bit, offset

Operation: if (!C) then PC <= PC + offset

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	JNC bit, offset	01010000 address offset	6

Notes:

1. The PC value used in the address calculation is the address of the next instruction.

JNZ

Jump if Accumulator Not Zero

JNZ bit, offset

Operation: if (A != 8'h00) then PC <= PC + offset

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	JNZ bit, offset	01110000 address offset	6

Notes:

1. The PC value used in the address calculation is the address of the next instruction.

JZ

Jump if Accumulator Zero

JZ bit, offset

Operation: if (A == 8'h00) then PC <= PC + offset

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	JZ bit, offset	01100000 address offset	6

Notes:

1. The PC value used in the address calculation is the address of the next instruction.

LCALL

Long Call

LCALL addr16

Operation:
(SP+1) <= PCL
(SP+2) <= PCH
PC <= addr16
SP <= SP+2

Flags:
C: Unaffected
AC: Unaffected
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
DA:	LCALL addr16	00010010 addr[15:8] addr[7:0]	6

LJMP

Long Jump

LJMP addr16

Operation: PC <= addr16

Flags:
C: Unaffected
AC: Unaffected
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
DA:	LJMP addr16	00000010 addr[15:8] addr[7:0]	6

MOV

Move Byte to Accumulator

MOV A, src — src: R, DA, IR, IM

Operation: A <= src

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
R:	MOV A, Rn	11101rrr	2
DA:	MOV A, direct	11100101 direct address	4
IR:	MOV A, @Ri	1110011i	4
IM:	MOV A, #data	01110100 immediate data	4

MOV

Move Byte to Register

MOV Rn, src src: A, DA, IM

Operation: Rn <= src

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
A:	MOV Rn, A	11111rrr	2
DA:	MOV Rn, direct	10101rrr direct address	4
IM:	MOV Rn, #data	01111rrr immediate data	4

MOV

Move Byte to Direct Address

MOV direct, src src: A, R, DA, IR, IM

Operation: direct <= src

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
A:	MOV direct, A	11110101 direct address	4
R:	MOV direct, Rn	10001rrr direct address	4
DA:	MOV direct, direct	10000101 direct src address direct dst address	6
IR:	MOV direct, @Ri	1000011i direct address	4
IM:	MOV direct, #data	01110101 direct address immediate data	6

MOV

Move Byte via Indirect Register

MOV @Ri, src src: A, DA, IM

Operation: @Ri <= src

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
A:	MOV @Ri, A	1111011i	4
DA:	MOV @Ri, direct	1010011i direct address	4
IM:	MOV @Ri, #data	0111011i immediate data	4

MOV

Move Bit

MOV dst, src

src: bit, C
dst: bit, C

Operation: dst <= src

Flags: **C:** Updated if destination.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
C, bit:	MOV C, bit	10100010 bit address	4
bit, C:	MOV bit, C	10010010 bit address	4

Notes:

1. This is a read-modify-write instruction, so when reading a Port register the data returned by the Port register is the contents of the output register rather than the input register.

MOV

Move to Data Pointer

MOV DPTR, src src: IM

Operation: DPTR <= src

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
IM:	MOV DPTR, imm16	10010000 imm16[15:8] imm16[7:0]	6

MOVC

Move Code Byte

MOVC A, src — src: DPTR-relative, PC-relative

Operation: A <= src

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
DPTRrel;	MOVC A, @A+DPTR	10010011	6
PCrel:	MOVC A, @A+PC	10000011	6

Notes:

1. The PC value used in the address calculation is the address of the next instruction.

MOVX

Move from External Byte

MOVX A, src src: DPTR, IR

Operation: A <= src

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
DPTR;	MOVX A, @DPTR	11100000	6
IR:	MOVX A, @Ri	1110001i	6

Notes:

1. External RAM uses the same address and data bus as Program Memory, but different data strobe signals.

2. When using register-indirect addressing, the upper byte of the address the contents of the Port 2 data output register.

MOVX

Move to External Byte

MOVX dst, A dst: DPTR, IR

Operation: dst <= A

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
DPTR	MOVX @DPTR, A	11110000	6
IR	MOVX @Ri, A	1111001i	6

Notes:

1. External RAM uses the same address and data bus as Program Memory, but different data strobe signals.

2. When using register-indirect addressing, the upper byte of the address the contents of the Port 2 data output register.

MUL

Multiply

MUL AB

Operation: {B, A} <= A * B

Flags: **C:** Cleared.
AC: Unaffected.
OV: Set if B = 0 after operation; cleared otherwise.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	MUL AB	10100100	4

Notes:

1. A and B are treated as unsigned numbers for the multiplication.

NOP

No Operation

NOP

Operation: none

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	NOP	00000000	2

ORL

Logical OR

ORL A, src — src: R, DA, IR, IM

Operation: A <= A | src

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
R:	ORL A, Rn	01001rrr	2
DA:	ORL A, direct	01000101 direct address	4
IR:	ORL A, @Ri	0100011i	4
IM:	ORL A, #data	01000100 immediate data	4

ORL

Logical OR with Memory

ORL direct, src src: A, IM

Operation: direct <= direct | src

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
A	ORL direct, A	01000010 direct address	4
IM	ORL direct, #data	01000011 direct address immediate data	6

Notes:

1. This is a read-modify-write instruction, so when reading a Port register the data returned by the Port register is the contents of the output register rather than the input register.

ORL

Logical OR with Carry

ORL C, src src: bit, /bit

Operation: C <= C | src, where src can be either a bit or the complement of a bit

Flags: **C:** Set/cleared with the result of the operation.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
Bit:	ORL C, bit	01110010 bit address	4
Bit:	ORL C, /bit	10100000 bit address	4

POP

Pop from Stack

POP direct

Operation: direct <= @SP
SP <= SP - 1

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	POP direct	11010000 direct address	4

PUSH

Push to Stack

PUSH direct

Operation: SP <= SP + 1
@SP <= direct

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	PUSH direct	11000000 direct address	4

RET

Return from Subroutine

RET

Operation: PC[15:8] <= @SP
PC[7:0] <= @SP - 1
SP <= SP -2

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	RET	00100010	6

RETI

Return from Interrupt

RETI

Operation: PC[15:8] <= @SP
PC[7:0] <= @SP - 1
SP <= SP -2

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	RETI	00110010	6

RL

Rotate Left

RL A

Operation: A <= {A[6:0], A[7]}

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	RL A	00100011	2

RLC

Rotate Left through Carry

RLC A

Operation: {C, A} <= {A[7:0], C}

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	RLC A	00110011	2

RR

Rotate Right

RR A

Operation: A <= {A[0], A[7:1]}

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	RR A	00000011	2

RRC

Rotate Right through Carry

RRC A

Operation: {C, A} <= {A[0], C, A[7:1]}

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	RRC A	00010011	2

SETB

Set (Bit)

SETB dst dst: C, bit

Operation: dst <= 1'b1

Flags: **C:** Set if dst is C; unaffected otherwise.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
C:	SETB C	11010011	2
Bit:	SETB bit	11010010 bit address	4

Notes:

1. This is a read-modify-write instruction, so when reading a Port register the data returned by the Port register is the contents of the output register rather than the input register.

SJMP

Short Jump

SJMP offset

Operation: PC <= PC + offset

Flags: **C:** Unaffected
AC: Unaffected
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	SJMP rel	10000000 address offset	6

Notes:

1. The PC value used in the address calculation is the address of the next instruction.

2. The relative address is sign-extended to 16 bits before the addition.

SUBB

Subtract

SUBB A, src src: R, DA, IR, IM

Operation: A <= A - src - C

Flags: **C:** Set if arithmetic borrow out of bit 7; cleared otherwise.
AC: Set if arithmetic borrow out of bit 3; cleared otherwise.
OV: Set if arithmetic overflow; cleared otherwise.

Addressing Modes	Assembly Syntax	Encoding	Clocks
R:	SUBB A, Rn	10011rrr	2
DA:	SUBB A, direct	10010101 direct address	4
IR:	SUBB A, @Ri	1001011i	4
IM:	SUBB A, #data	10010100 immediate data	4

SWAP

Swap Nibbles

SWAP A

Operation: A <= {A[3:0], A[7:4]}

Flags: **C:** Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
	SWAP A	11000100	2

XCH

Exchange Bytes

XCH A, src — src: R, DA, IR

Operation:
tmp <= src
src <= A
A <= tmp

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
R:	XCH A, Rn	11001rrr	2
DA:	XCH A, direct	11000101 direct address	4
IR:	XCH A, @Ri	1100011i	4

XCHD

Exchange Digit

XCHD A, src src: IR

Operation:
tmp <= src[3:0]
src[3:0] <= A[3:0]
A[3:0] <= tmp

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
IR:	XCHD A, @Ri	1101011i	4

XRL

Logical XOR

XRL A, src — src: R, DA, IR, IM

Operation: A <= A ^ src

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
R:	XRL A, Rn	01101rrr	2
DA:	XRL A, direct	01100101 direct address	4
IR:	XRL A, @Ri	0110011i	4
IM:	XRL A, #data	01100100 immediate data	4

XRL

Logical XOR with Memory

XRL direct, src — src: A, IM

Operation: direct <= direct ^ src

Flags:
C: Unaffected.
AC: Unaffected.
OV: Unaffected.

Addressing Modes	Assembly Syntax	Encoding	Clocks
A	XRL direct, A	01100010 direct address	4
IM	XRL direct, #data	01100011 direct address immediate data	6

Notes:

1. This is a read-modify-write instruction, so when reading a Port register the data returned by the Port register is the contents of the output register rather than the input register.

Special Function Registers

The table below lists all of the internal Special Function Registers used in the design, along with their mnemonics, address, bit addressability and reset state. SFR addresses not listed here are considered external, and will be accessed via the External SFR bus.

Register Name	Mnemonic	Address	R/W	Bit	Reset
Port 0	P0	0x80	R & W	Bit	11111111
Stack Pointer	SP	0x81	R/W		00011111
Data Pointer LSB	DPL	0x82	R/W		00000000
Data Pointer MSB	DPH	0x83	R/W		00000000
Data Pointer 1 LSB	DP1L	0x84	R/W		00000000
Data Pointer 1 MSB	DP1H	0x85	R/W		00000000
Data Pointer Select	DPS	0x86	R/W		00000000
Power Control	PCON	0x87	R/W		00000000
Timer Control	TCON	0x88	R/W	Bit	00000000
Timer Mode	TMOD	0x89	R/W		00000000
Timer 0 Time Constant LSB	TL0	0x8A	R/W		00000000
Timer 0 Time Constant MSB	TH0	0x8B	R/W		00000000
Timer 1 Time Constant LSB	TL1	0x8C	R/W		00000000
Timer 1 Time Constant MSB	TH1	0x8D	R/W		00000000
RAM Control	RCON	0x8E	R/W		00000000
Extended Control	ECON	0x8F	R/W		00000000
Port 1	P1	0x90	R & W	Bit	11111111
Serial Control	SCON	0x98	R/W	Bit	00000000
Serial Buffer	SBUF	0x99	R & W		xxxxxxxx
Baud Rate Reload LSB	BRLL	0x9A	R/W		00000000
Baud Rate Reload MSB	BRLH	0x9B	R/W		00000000
Baud Rate Control	BRCON	0x9C	R/W		00000000
Extended Serial Control	ESCON	0x9D	R/W		00000000
Slave Address	SADDR	0x9E	R/W		00000000
Slave Address Mask	SADEN	0x9F	R/W		00000000

Port 2	P2	0xA0	R & W	Bit	11111111
Interrupt Enable	IE	0xA8	R/W	Bit	00000000
Port 3	P3	0xB0	R & W	Bit	11111111
Interrupt Priority 0	IP0	0xB8	R/W	Bit	00000000
Processor Status Word	PSW	0xD0	R/W	Bit	00000000
Accumulator	ACC	0xE0	R/W	Bit	00000000
B Register	B	0xF0	R/W	Bit	00000000
Interrupt Priority 1	IP1	0xF8	R/W	Bit	00000000

CPU Registers

A number of addresses in the internal Special Function Register address space are used for CPU registers, and are accessed implicitly by instructions. These registers can also be accessed using direct addressing.

Features:

- CPU registers are implemented using flip-flops for highest performance.

- Alternate Data Pointer to store an additional 16-bit address.

CPU Registers:

Register Name	Mnemonic	Address	R/W	Bit	Reset
Stack Pointer	SP	0x81	R/W		00011111
Data Pointer LSB	DPL	0x82	R/W		00000000
Data Pointer MSB	DPH	0x83	R/W		00000000
Data Pointer 1 LSB	DP1L	0x84	R/W		00000000
Data Pointer 1 MSB	DP1H	0x85	R/W		00000000
Data Pointer Select	DPS	0x86	R/W		00000000
Processor Status Word	PSW	0xD0	R/W	Bit	00000000
Accumulator	ACC	0xE0	R/W	Bit	00000000
B Register	B	0xF0	R/W	Bit	00000000

Register Descriptions

Stack Pointer		(SP) (Address = 0x81)
Bit(s)	**Value**	**Description**
7:0		The Stack Pointer holds the address of the stack in Internal RAM. The stack grows upward.

Data Pointer LSB		(DPL) (Address = 0x82)
Bit(s)	**Value**	**Description**
7:0		This byte holds the least-significant byte of the 16-bit Data Pointer. The Data Pointer is used to address locations in either Program memory or External RAM.

Data Pointer MSB		(DPH) (Address = 0x83)
Bit(s)	**Value**	**Description**
7:0		This byte holds the most-significant byte of the 16-bit Data Pointer. The Data Pointer is used to address locations in either Program memory or External RAM.

Data Pointer 1 LSB		(DP1L) (Address = 0x84)
Bit(s)	**Value**	**Description**
7:0		This byte holds the least-significant byte of the 16-bit Alternate Data Pointer. Only one of the Data Pointers can be active at any given time, selected by a bit in the Data Pointer Select register.

Data Pointer 1 MSB		(DP1H) (Address = 0x85)
Bit(s)	**Value**	**Description**
7:0		This byte holds the most-significant byte of the 16-bit Alternate Data Pointer. Only one of the Data Pointers can be active at any given time, selected by a bit in the Data Pointer Select register.

Data Pointer Select	(DPS)	(Address = 0x856
Bit(s)	**Value**	**Description**
7:1		These bits are reserved and will always be read as zero.
0	0	Select {DPH, DPL} as the Data Pointer for use by instructions.
	1	Select {DP1H, DP1L} as the Data Pointer for use by instructions.

Program Status Word	(PSW)	(Address = 0xD0)
Bit(s)	**Value**	**Description**
7		CPU Carry flag. This flag is used for arithmetic carry, shift linkage bit, and bit operation results.
6		CPU Alternate Carry flag. This flag is the carry out of the lower nibble, and is used for BCD math.
5		User Flag 0. This flag can bet set/reset using bit operation instructions.
4:3	00	Select Internal RAM addresses 0x00-0x07 to act as CPU registers R0-R7.
	01	Select Internal RAM addresses 0x08-0x0F to act as CPU registers R0-R7.
	10	Select Internal RAM addresses 0x10-0x17 to act as CPU registers R0-R7.
	11	Select Internal RAM addresses 0x18-0x1F to act as CPU registers R0-R7.
2		CPU Overflow flag. This bit is used to signal arithmetic overflow.
1		User Flag 1. This flag can bet set/reset using bit operation instructions.
0		CPU Parity flag. This flag is updated with the parity (0 = even, 1 = odd) of the Accumulator whenever the Accumulator is written by the CPU.

Accumulator	(ACC)	(Address = 0xE0)
Bit(s)	**Value**	**Description**
7:0		This byte is the Accumulator for the CPU.

B	(B)	(Address = 0xF0)
Bit(s)	**Value**	**Description**
7:0		This byte is the B register for the CPU, used with the Multiply and Divide instructions.

System Registers

System Registers are used to configure the hardware of the design. Only one of these registers is used inside the design, for selecting banks of RAM. The other register is merely output by the design for use externally.

Features:

- Uncommitted 8-bit Power Control register output for user-defined functionality external to the CPU core.

- Up to sixteen banks for Internal RAM addresses, when using FPGA RAM macros.

System Registers

Register Name	Mnemonic	Address	R/W	Bit	Reset
Power Control	PCON	0x87	R/W		00000000
Internal RAM Control	RCON	0x8E	R/W		00000000
Extended Control	ECON	0x8F	R/W		00000000

Register Descriptions

Power Control		(PCON (Address = 0x87)
Bit(s)	**Value**	**Description**
7:0		The Power Control register is output by the Y51 design as the **pcon_reg** bus for external use.

RAM Control		(RCON) (Address = 0x8E)
Bit(s)	**Value**	**Description**
7:4		These bits are reserved.
3:0		These bits select the bank for the upper half of the Internal Ram address space (addresses 0x80-0xFF). The number of banks available depends on the target technology. In the case of a Microsemi A3P target, four banks are available. In the case of a Lattice ICE40 or Xilinx Spartan-3 target, sixteen banks are available.

Extended Control		(ECON (Address = 0x8F)
Bit(s)	**Value**	**Description**
7:0		The Extended Control register is output by the Y51 design as the **econ_reg** bus for external use.

Interrupt Control

Interrupt Control manages all of the on-chip and external interrupt requests. Interrupt control automatically prioritizes interrupt requests and generates interrupt vectors.

Features:

- Five standard 8051 interrupt sources, plus two additional internal interrupt inputs.

- Global interrupt enable.

- Individual enables for interrupt requests.

- Default priority is: (highest to lowest) External 0, Timer 0 Overflow, External 1, Timer 1 Overflow, Serial Port, Internal 0 and Internal 1.

- Four programmable priority levels for each interrupt request allows customizing the overall interrupt priority.

- Standard 8051 interrupt vectors: 0x0003 (External 0), 0x000B (Timer 0 Overflow), 0x0013 (External 1), 0x001B (Timer 1 Overflow), 0x0023 (Serial Port), plus 0x002B (Internal 0) and 0x0033 (Internal 1).

Interrupt Control Registers

Register Name	Mnemonic	Address	R/W	Bit	Reset
Interrupt Enable	IE	0xA8	R/W	Bit	00000000
Interrupt Priority 0	IP0	0xB8	R/W	Bit	00000000
Interrupt Priority 1	IP1	0xF8	R/W	Bit	00000000

Register Descriptions

Interrupt Enable (IE) (Address = 0xA8		
Bit(s)	**Value**	**Description**
7	0	All interrupts are disabled.
	1	Interrupts are globally enabled.
6	0	Disable Internal 1 interrupt.
	1	Enable Internal 1 interrupt.
5	0	Disable Internal 0 interrupt.
	1	Enable Internal 0 interrupt.
4	0	Disable Serial Port interrupt.
	1	Enable Serial Port interrupt.
3	0	Disable Timer 1 Overflow interrupt.
	1	Enable Timer 1 Overflow interrupt.
2	0	Disable External 1 interrupt.
	1	Enable External 1 interrupt.
1	0	Disable Timer 0 Overflow interrupt.
	1	Enable Timer 0 Overflow interrupt.
0	0	Disable External 0 Interrupt.
	1	Enable External 0 Interrupt.

Interrupt Priority 0 (IP0) (Address = 0xB0		
Bit(s)	**Value**	**Description**
7		Reserved.
6		LSB of interrupt priority for Internal 1 interrupt.
5		LSB of interrupt priority for Internal 0 interrupt.
4		LSB of interrupt priority for Serial Port interrupt.
3		LSB of interrupt priority for Timer 1 Overflow interrupt.
2		LSB of interrupt priority for External 1 interrupt.
1		LSB of interrupt priority for Timer 0 Overflow interrupt.
0		LSB of interrupt priority for External 0 interrupt.

Interrupt Priority 1 (IP1) (Address = 0xF0		
Bit(s)	**Value**	**Description**
7		Reserved
6		MSB of interrupt priority for Internal 1 interrupt.
5		MSB of interrupt priority for Internal 0 interrupt.
4		MSB of interrupt priority for Serial Port interrupt.
3		MSB of interrupt priority for Timer 1 Overflow interrupt.
2		MSB of interrupt priority for External 1 interrupt.
1		MSB of interrupt priority for Timer 0 Overflow interrupt.
0		MSB of interrupt priority for External 0 interrupt.

Parallel Ports

This CPU design contains four simple parallel ports. For maximum compatibility, no multiplexing of port signals with other peripheral signals is done internally in the design.

Features:

- Four simple parallel ports.

- Separate input and output registers.

- Correctly implements read-modify-write operation, where read returns the output register instead of the input register.

Registers

Register Name	Mnemonic	Address	R/W	Bit	Reset
Port 0	P0	0x80	R & W	Bit	11111111
Port 1	P1	0x90	R & W	Bit	11111111
Port 2	P2	0xA0	R & W	Bit	11111111
Port 3	P3	0xB0	R & W	Bit	11111111

Register Descriptions

Port 0 (P0) (Address = 0x80)		
Bit(s)	**Value**	**Description**
7:0	read	Returns the state of the input register. In the case of a read-modify-write instruction, the contents of the output register are returned.
	write	Loads the output register.

Port 1 (P1) (Address = 0x90)		
Bit(s)	**Value**	**Description**
7:0	read	Returns the state of the input register. In the case of a read-modify-write instruction, the contents of the output register are returned.
	write	Loads the output register.

Port 2 (P2) (Address = 0xA0)		
Bit(s)	**Value**	**Description**
7:0	read	Returns the state of the input register. In the case of a read-modify-write instruction, the contents of the output register are returned.
	write	Loads the output register.

Port 3 (P3) (Address = 0xB0)		
Bit(s)	**Value**	**Description**
7:0	read	Returns the state of the input register. In the case of a read-modify-write instruction, the contents of the output register are returned.
	write	Loads the output register.

Timer /Counters

Two general-purpose 16-bit timer/counters, with multiple operating modes.

Features:

- Two 8-bit or 16-bit timer/counters

- Clocked at **clkc**/12 rate for backwards timing compatibility.

- Separate count and gate inputs for each timer/counter.

Registers

Register Name	Mnemonic	Address	R/W	Bit	Reset
Timer Control	TCON	0x88	R/W	Bit	00000000
Timer Mode	TMOD	0x89	R/W		00000000
Timer 0 Time Constant LSB	TL0	0x8A	R/W		00000000
Timer 1 Time Constant LSB	TL1	0x8B	R/W		00000000
Timer 0 Time Constant MSB	TH0	0x8C	R/W		00000000
Timer 1 Time Constant MSB	TH1	0x8D	R/W		00000000

Register Descriptions

Timer Control (TCON) (Address = 0x88)		
Bit(s)	**Value**	**Description**
7	0	Automatically cleared when the CPU vectors to address 0x001B to service a Timer 1 overflow interrupt.
	1	Automatically set when the Timer 1 count overflows.
6	0	Disable Timer 1.
	1	Enable Timer 1.
5	0	Automatically cleared when the CPU vectors to address 0x000B to service a Timer 0 overflow interrupt.
	1	Automatically set when the Timer 0 count overflows.
4	0	Disable Timer 0.
	1	Enable Timer 0.
3		When the External 1 interrupt is level-sensitive, this bit directly reflects the state of the **ext1_int** signal. When the External 1 interrupt is edge-triggered, this bit is set by a rising edge on the **ext1_int** signal, and remains set until the CPU vectors to address 0x0013.
2	0	Generate External 0 interrupt while **ext1_int** signal is High.
	1	Generate External 0 interrupt on rising edge on **ext1_int** signal.
1		When the External 0 interrupt is level-sensitive, this bit directly reflects the state of the **ext0_int** signal. When the External 0 interrupt is edge-triggered, this bit is set by a rising edge on the **ext0_int** signal, and remains set until the CPU vectors to address 0x0003.
0	0	Generate External 0 interrupt while **ext0_int** signal is High.
	1	Generate External 0 interrupt on rising edge on **ext0_int** signal.

Timer Mode		(TMOD) (Address = 0x89)
Bit(s)	**Value**	**Description**
7	0	The **t1_gate** signal has no effect on Timer 1 operation.
	1	Enable Timer 1 to count only while the **t1_gate** signal is High.
6	0	Timer 1 increments on the **clk** signal divided by 12.
	1	Timer 1 increments on each falling edge on the **t1_cnt** signal.
5:4	00	Mode 0: 8-bit counter (using TH1), with a 5-bit prescalar (TL1).
	01	Mode 1: 16-bit timer/counter.
	10	Mode 2: 8-bit auto-reload timer/counter (using TL1). Reloaded on overflow from TH1.
	11	Mode 3: Timer 1 is halted, but retains the current count.
3	0	The **t0_gate** signal has no effect on Timer 0 operation.
	1	Enable Timer 1 to count only while the **t1_gate** signal is High.
2	0	Timer 0 increments on the **clk** signal divided by 12.
	1	Timer 0 increments on each falling edge on the **t0_cnt** signal.
1:0	00	Mode 0: 8-bit counter (using TH0), with a 5-bit prescalar (TL0).
	01	Mode 1: 16-bit timer/counter.
	10	Mode 2: 8-bit auto-reload timer/counter (using TL0). Reloaded on overflow from TH0.
	11	Mode 3: 8-bit timer/counter (using TL0). additional 8-bit timer/counter (using TH0). In this mode TCON[7] and TCON[6] are controlled by the TH0 timer/ counter rather than by Timer 1. Timer 1 is still available.

Timer 0 LSB		(TL0) (Address = 0x8A)
Bit(s)	**Value**	**Description**
7:0		Least-significant byte of Timer 0 count.

Timer 1 LSB		(TL1) (Address = 0x8B)
Bit(s)	**Value**	**Description**
7:0		Least-significant byte of Timer 1 count.

Timer 1 LSB		(TH0) (Address = 0x8C)
Bit(s)	**Value**	**Description**
7:0		Most-significant byte of Timer 0 count.

Timer 1 MSB		(TH1) (Address = 0x8D)
Bit(s)	**Value**	**Description**
7:0		Most-significant byte of Timer 1 count.

Serial Interface

The Serial Interface port provides basic full-duplex async serial communication and half-duplex clocked serial communication. Backwards-compatible enhancements provide improved error handling, a dedicated baud rate generator, and multiprocessor address recognition in hardware.

Features:

- Full-duplex async, 8 bits/character, clocked at 16x the data rate

- Half-duplex synchronous serial, clocked at **clkc**/12 rate

- Optional enhanced error reporting

- Optional enhanced Multiprocessor Mode with address recognition

- Optional enhanced clocking from Timers or dedicated baud-rate generator

Registers

Register Name	Mnemonic	Address	R/W	Bit	Reset
Serial Control	SCON	0x98	R/W		00000000
Serial Buffer	SBUF	0x99	R/W		xxxxxxxx
Baud Rate Reload LSB	BRLL	0x9A	R/W		00000000
Baud Rate Reload MSB	BRLH	0x9B	R/W		00000000
Baud Rate Control	BRCON	0x9C	R/W		00000000
Extended Serial Control	ESCON	0x9D	R/W		00000000
Slave Address	SADDR	0x9E	R/W		00000000
Slave Address Mask	SADEN	0x9F	R/W		00000000

Register Descriptions

Serial Control (Basic Mode) (SCON) (Address = 0x98)		
Bit(s)	**Value**	**Description**
7:6	00	Mode 0: half-duplex shift register, clocked at **clkc**/12 rate
	01	Mode 1: 8-bit UART, clocked by Timer 1 output or internal BRG
	10	Mode 2: 9-bit UART, clocked at **clkc**/2 rate
	11	Mode 3: 9-bit UART, clocked by Timer 1 output or internal BRG
5	0	Disable multiprocessor communication mode.
	1	Enable multiprocessor communication mode. Receiver only accepts bytes with the ninth bit set to 1 and an address match. Only applies in UART modes.
4	0	Disable receiver.
	1	Enable receiver. In Mode 0 reception is initiated when this bit is set while the Receive Interrupt Pending bit is cleared.
3		Value to be transmitted as ninth (address) bit in 9-bit UART modes. Ignored in other modes.
2 (rd-only)		Value of received ninth (address) bit in 9-bit UART modes. Always zero in other modes.
1	0	No transmit interrupt pending. Write with 0 to clear transmit interrupt.
	1	Transmit interrupt pending. Automatically set when the transmitter has finished sending a byte.
0	0	No receive interrupt pending. Write with 0 to clear receive interrupt.
	1	Receive interrupt pending. Automatically set when the receiver writes to the receive buffer. In Mode 0 reception is initiated when this bit is cleared while the receiver enable bit is set.

Serial Control (Enhanced Mode) (SCON) (Address = 0x98)		
Bit(s)	**Value**	**Description**
7	0	No Framing Error.
	1	Framing Error on the current receive byte.
6	0	No Overrun Error.
	1	Overrun Error. This bit is automatically cleared by the read of this register.
5	0	Disable multiprocessor communication mode.
	1	Enable multiprocessor communication mode. Receiver only accepts bytes with the ninth bit set to 1 and an address match. Only applies in UART modes.
4	0	Disable receiver.
	1	Enable receiver. In Mode 0 reception is initiated when this bit is set while the Receive Interrupt Pending bit is cleared.
3		Value to be transmitted as ninth (address) bit in 9-bit UART modes. Ignored in other modes.
2 (rd-only)		Value of received ninth (address) bit in 9-bit UART modes. Always 0 in other modes.
1	0	No transmit interrupt pending. Write with 0 to clear transmit interrupt.
	1	Transmit interrupt pending. Automatically set when the transmitter has finished sending a byte.
0	0	No receive interrupt pending. Write with 0 to clear receive interrupt.
	1	Receive interrupt pending. Automatically set when the receiver writes to the receive buffer. In Mode 0 reception is initiated when this bit is cleared while the receiver enable bit is set.

Serial Buffer (SBUF) (Address = 0x99)		
Bit(s)	**Value**	**Description**
7:0	Read	Returns the received byte.
	Write	Loads a byte for transmission. In Mode 0, writing to the Serial Buffer initiates transmission.

Baud Rate Reload LSB (BRLL) (Address = 0x9A)		
Bit(s)	**Value**	**Description**
7:0		LSB of Reload value for Baud Rate Generator. Automatically loaded on the next serial clock when the baud rate counter reaches the count of 0xFFFF.

Baud Rate Reload MSB (BRLH) (Address = 0x9B)		
Bit(s)	**Value**	**Description**
7:0		MSB of Reload value for Baud Rate Generator. Automatically loaded on the next serial clock when the baud rate counter reaches the count of 0xFFFF.

Baud Rate Control (BRCON) (Address = 0x9C)		
Bit(s)	**Value**	**Description**
7:5		These bits are reserved.
4	0	Disable Baud Rate Generator.
	1	Enable Baud Rate Generator.
3	0	Use Timer 1 output as transmit clock.
	1	Use Baud Rate Generator output as transmit clock.
2	0	Use Timer 1 output as receive clock.
	1	Use Baud Rate Generator output as receive clock.
1	0	Baud Rate Generator clock is system clock divided by 12.
	1	Baud Rate Generator clock is system clock divided by 2.
0	0	In Mode 0, use system clock (divided by 12) as serial clock.
	1	In Mode 0, use Baud Rate Generator output as serial clock.

Extended Serial Control (ESCON) (Address = 0x9D)		
Bit(s)	**Value**	**Description**
7:6	00	Mode 0: half-duplex shift register, clocked by **clkc**/12 or internal BRG.
	01	Mode 1: 8-bit UART, clocked by Timer 1 output or internal BRG
	10	Mode 2: 9-bit UART, clocked at **clkc**/2 rate
	11	Mode 3: 9-bit UART, clocked by Timer 1 output or internal BRG
5	0	Disable multiprocessor communication mode.
	1	Enable multiprocessor communication mode. Receiver only accepts bytes with the ninth bit set to 1 and an address match. Only applies in UART modes.
4		This bit is not used.
3	0	No effect on transmitter.
	1	Send Break in UART modes; ignored in Mode 0.
2:1	00	In Mode 0 clock output idles High, data sampled on rising edge
	01	In Mode 0 clock output idles Low, data sampled on rising edge
	10	In Mode 0 clock output idles Low, data sampled on falling edge
	11	In Mode 0 clock output idles High, data sampled on falling edge
0	0	Basic Serial Port operation, compatible with original 8051.
	1	Enhanced Serial Port operation. Redefines Serial Control register, and activates bits [7:6] in this register.

Slave Address (SADDR) (Address = 0x9E)		
Bit(s)	**Value**	**Description**
7:0		Multiprocessor mode address. If Multiprocessor communication mode is enabled only address bytes (marked with a one in the ninth bit) matching this address will be accepted by the receiver.

Slave Address Mask (SADEN) (Address = 0x9F)		
Bit(s)	**Value**	**Description**
7:0		Multiprocessor mode address mask. If a bit in this register is set to one, then in Multiprocessor communication mode only address bytes (marked with a one in the ninth bit) matching the address in that bit position will be accepted by the receiver. All zeros in this register disables address-based filtering.