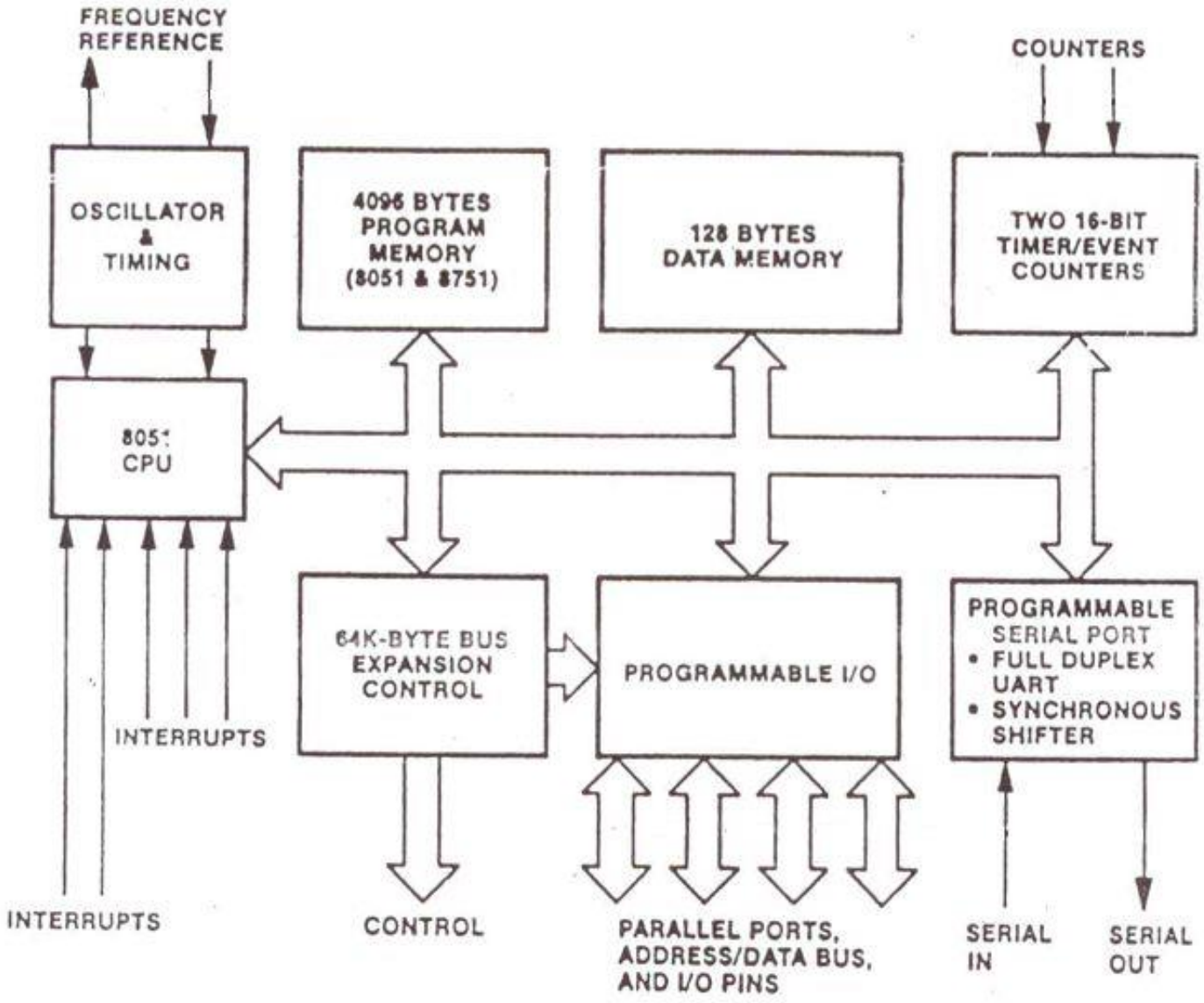
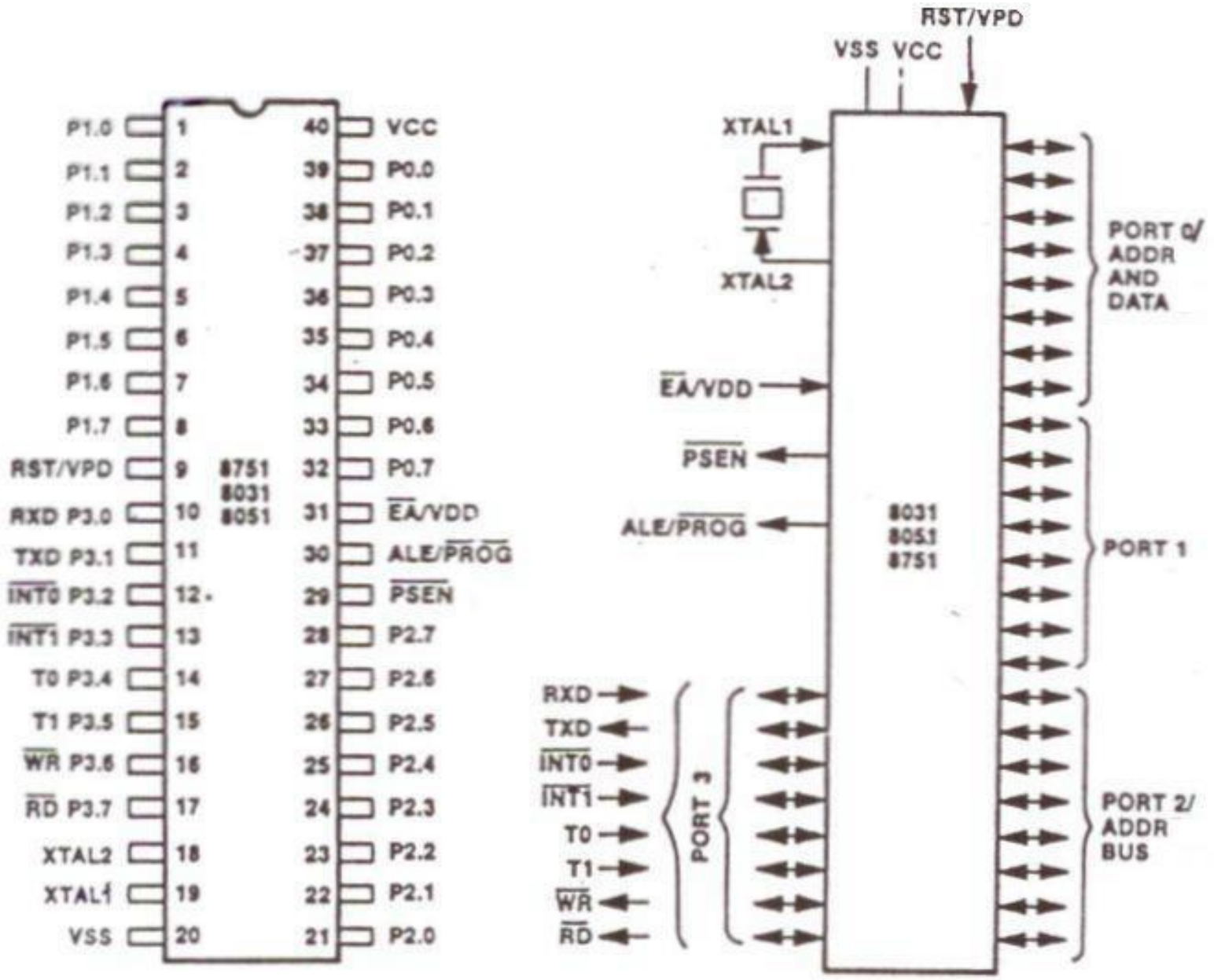
JEDNOČIPOVÉ MIKROPOČÍTAČE

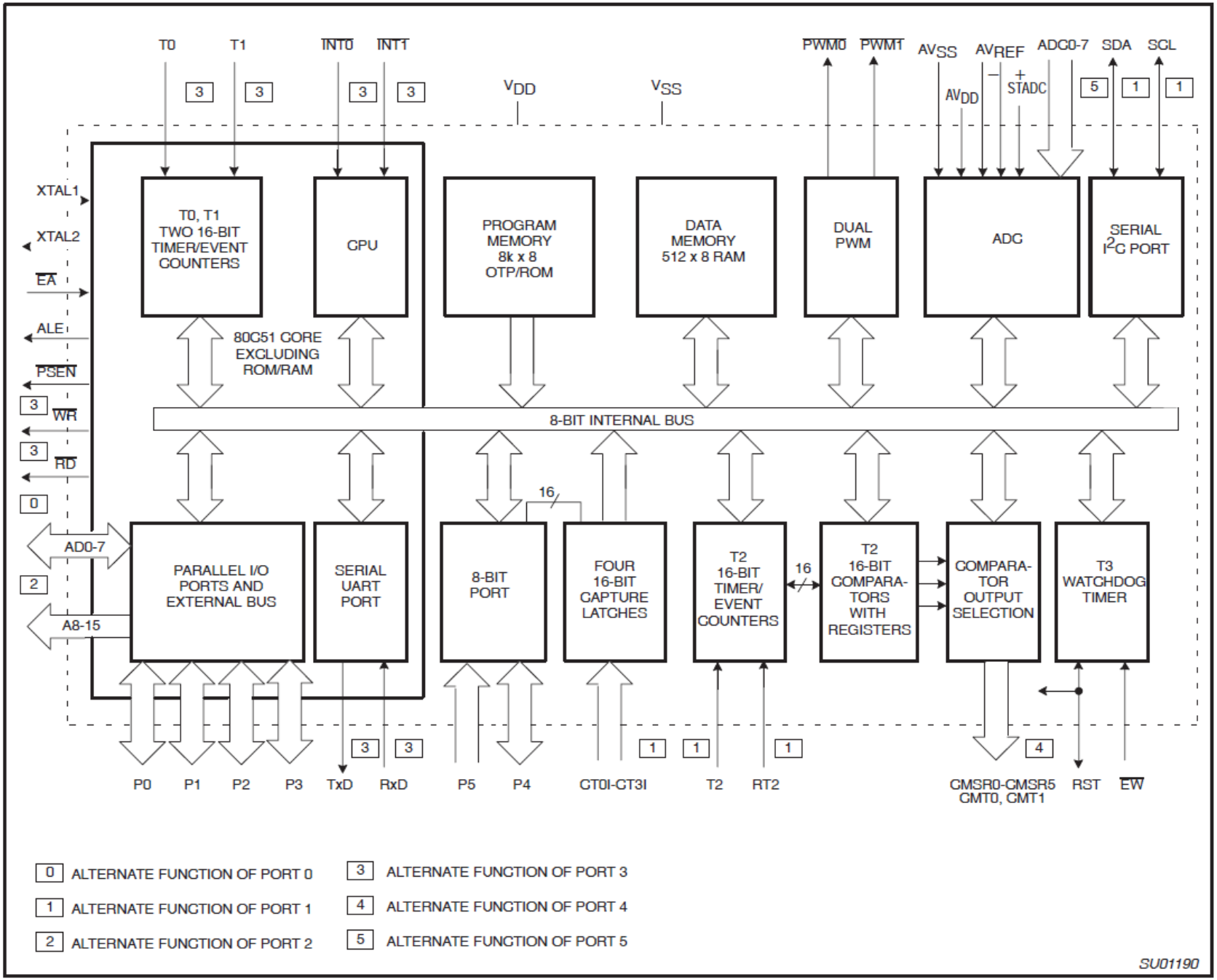
ONE-CHIP MICROCOMPUTERS (MICROCONTROLLERS)

RODINA x51


FREQUENCY REFERENCE
COUNTERS
OSCILLATOR & TIMING
4096 BYTES PROGRAM MEMORY (8051 & 8751)
128 BYTES DATA MEMORY
TWO 16-BIT TIMER/EVENT COUNTERS
8051 CPU
64K-BYTE BUS EXPANSION CONTROL
PROGRAMMABLE I/O
PROGRAMMABLE SERIAL PORT
• FULL DUPLEX UART
• SYNCHRONOUS SHIFTER
INTERRUPTS
INTERRUPTS
CONTROL
PARALLEL PORTS, ADDRESS/DATA BUS, AND I/O PINS
SERIAL IN
SERIAL OUT


P1.0 1
P1.1 2
P1.2 3
P1.3 4
P1.4 5
P1.5 6
P1.6 7
P1.7 8
RST/VPD 9
RXD P3.0 10
TXD P3.1 11
INT0 P3.2 12
INT1 P3.3 13
T0 P3.4 14
T1 P3.5 15
WR P3.6 16
RD P3.7 17
XTAL2 18
XTAL1 19
VSS 20
8751
8031
8051
40 VCC
39 P0.0
38 P0.1
37 P0.2
36 P0.3
35 P0.4
34 P0.5
33 P0.6
32 P0.7
31 EA/VDD
30 ALE/PROG
29 PSEN
28 P2.7
27 P2.6
26 P2.5
25 P2.4
24 P2.3
23 P2.2
22 P2.1
21 P2.0
RST/VPD
VSS VCC
XTAL1
XTAL2
EA/VDD
PSEN
ALE/PROG
8031
8051
8751
PORT 0/ ADDR AND DATA
PORT 1
PORT 2/ ADDR BUS
PORT 3
RXD
TXD
INT0
INT1
T0
T1
WR
RD

I_{CC}	Power Supply Current:		
	Active Mode		
	at 12 MHz (Figure 5)	30	mA
	at 16 MHz	38	mA
	at 24 MHz	56	mA
	at 33 MHz (8XC5X-33)	56	mA
	Idle Mode		
	at 12 MHz (Figure 5)	7.5	mA
	at 16 MHz	9.5	mA
	at 24 MHz	13.5	mA
	at 33 MHz (8XC5X-33)	15	mA
	Power Down Mode	75	μA
	8XC5X-33	50	μA



0 ALTERNATE FUNCTION OF PORT 0

1 ALTERNATE FUNCTION OF PORT 1

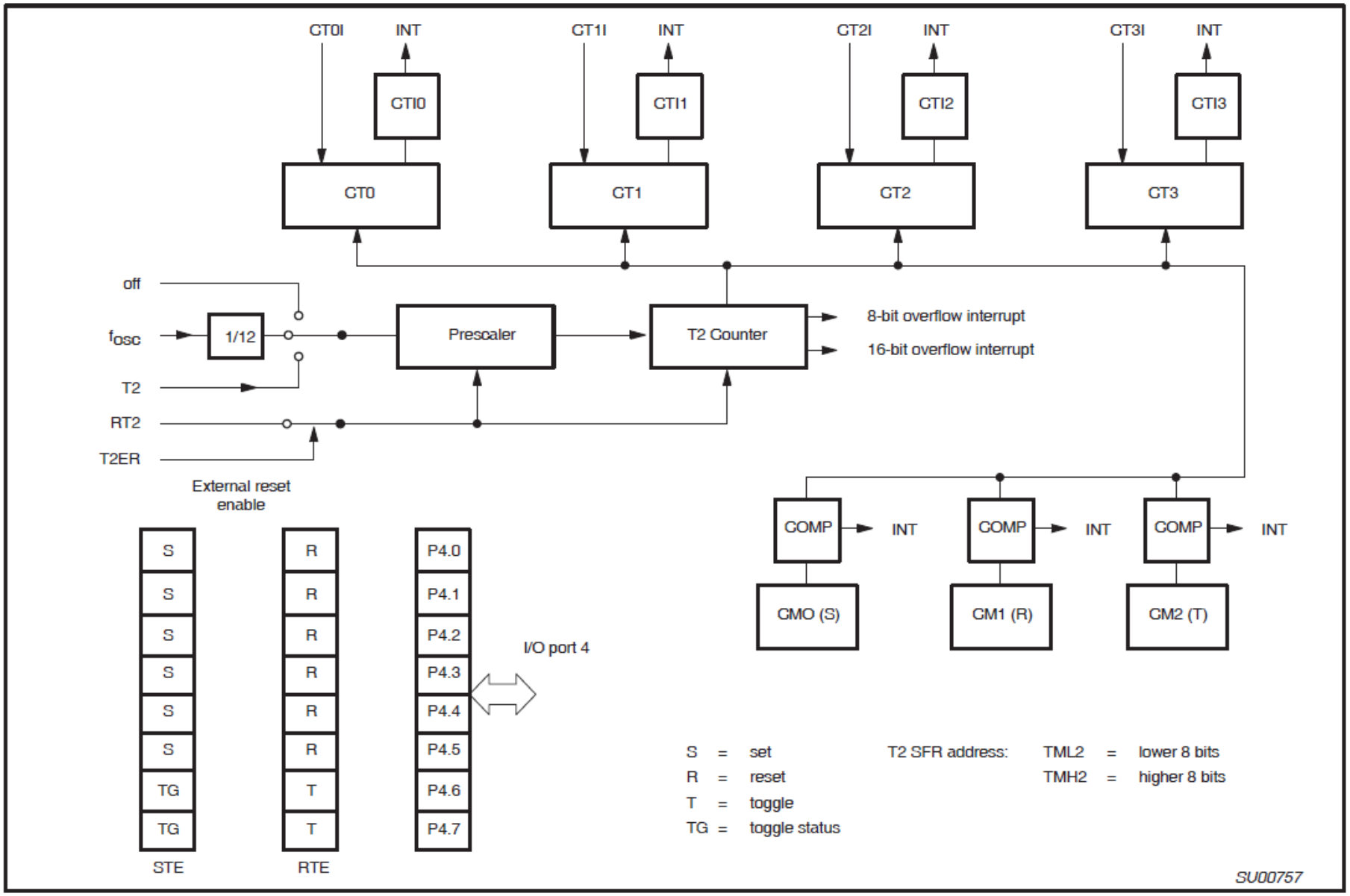
2 ALTERNATE FUNCTION OF PORT 2

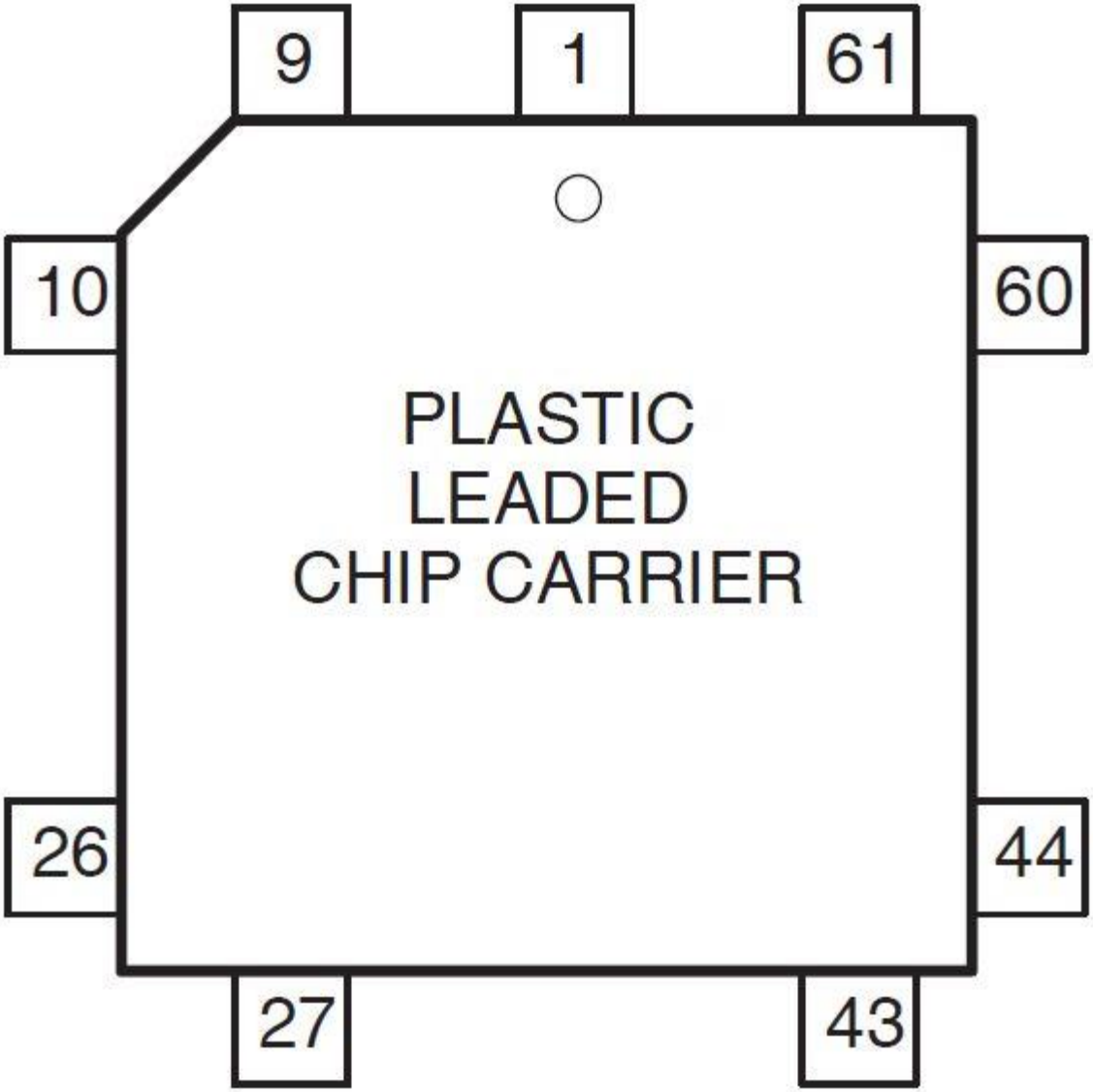
3 ALTERNATE FUNCTION OF PORT 3

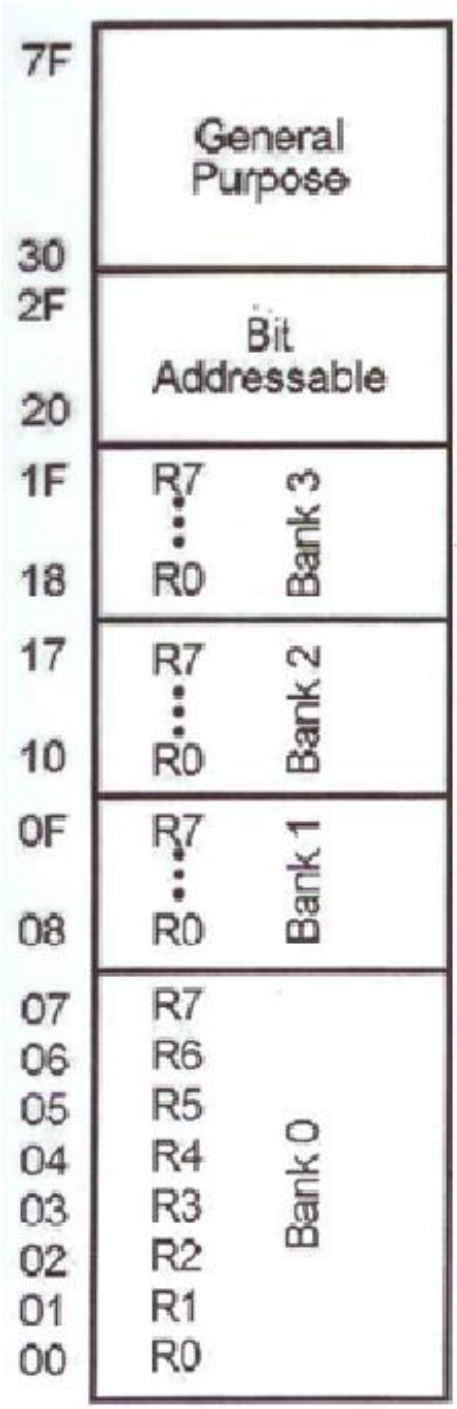
4 ALTERNATE FUNCTION OF PORT 4

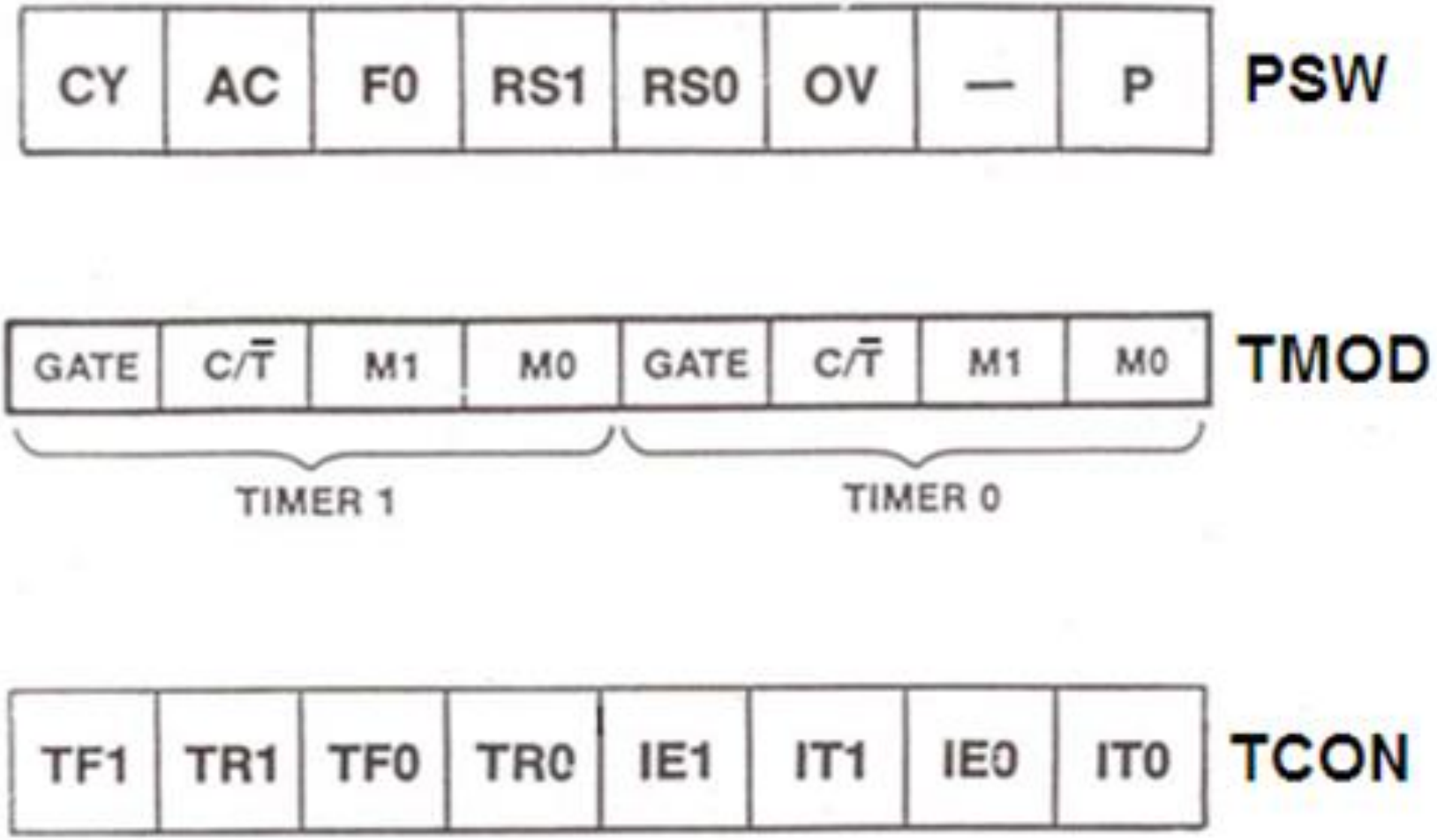
5 ALTERNATE FUNCTION OF PORT 5

SU01190


CT0I
INT
CTI0
CT0
CT1I
INT
CTI1
CT1
CT2I
INT
CTI2
CT2
CT3I
INT
CTI3
CT3
off
fosc
1/12
T2
RT2
T2ER
External reset enable
Prescaler
T2 Counter
8-bit overflow interrupt
16-bit overflow interrupt
COMP
INT
CM0 (S)
CM1 (R)
CM2 (T)
S
S
S
S
S
S
TG
TG
STE
R
R
R
R
R
R
T
T
RTE
P4.0
P4.1
P4.2
P4.3
P4.4
P4.5
P4.6
P4.7
I/O port 4
S = set
R = reset
T = toggle
TG = toggle status
T2 SFR address:
TML2 = lower 8 bits
TMH2 = higher 8 bits
SU00757


9
1
61
10
60
PLASTIC
LEADED
CHIP CARRIER
26
44
27
43

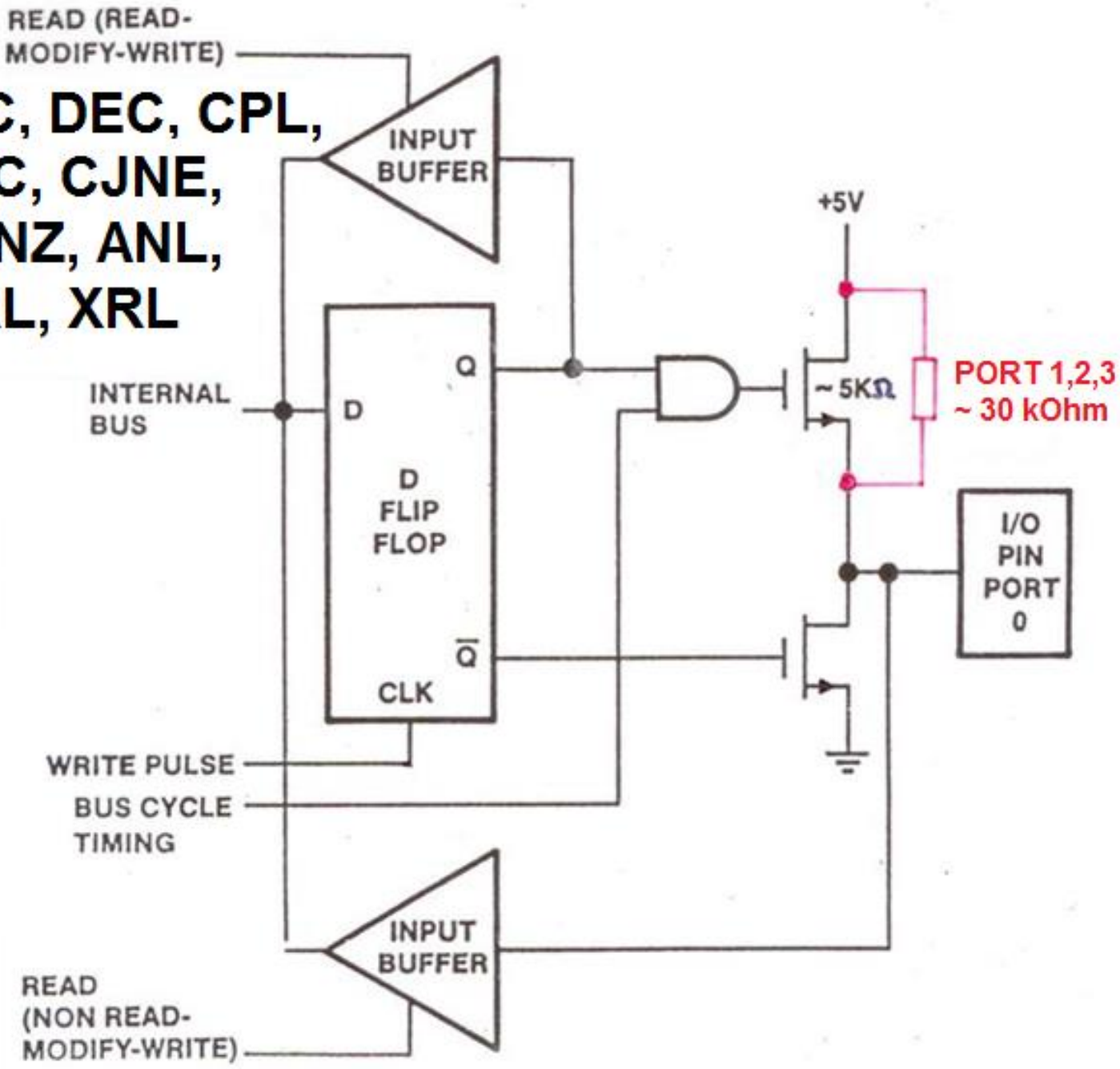

7F
General Purpose
30
2F
Bit Addressable
20
1F
R7
R0
Bank 3
18
17
R7
R0
Bank 2
10
0F
R7
R0
Bank 1
08
07
R7
06
R6
05
R5
04
R4
03
R3
02
R2
01
R1
00
R0
Bank 0

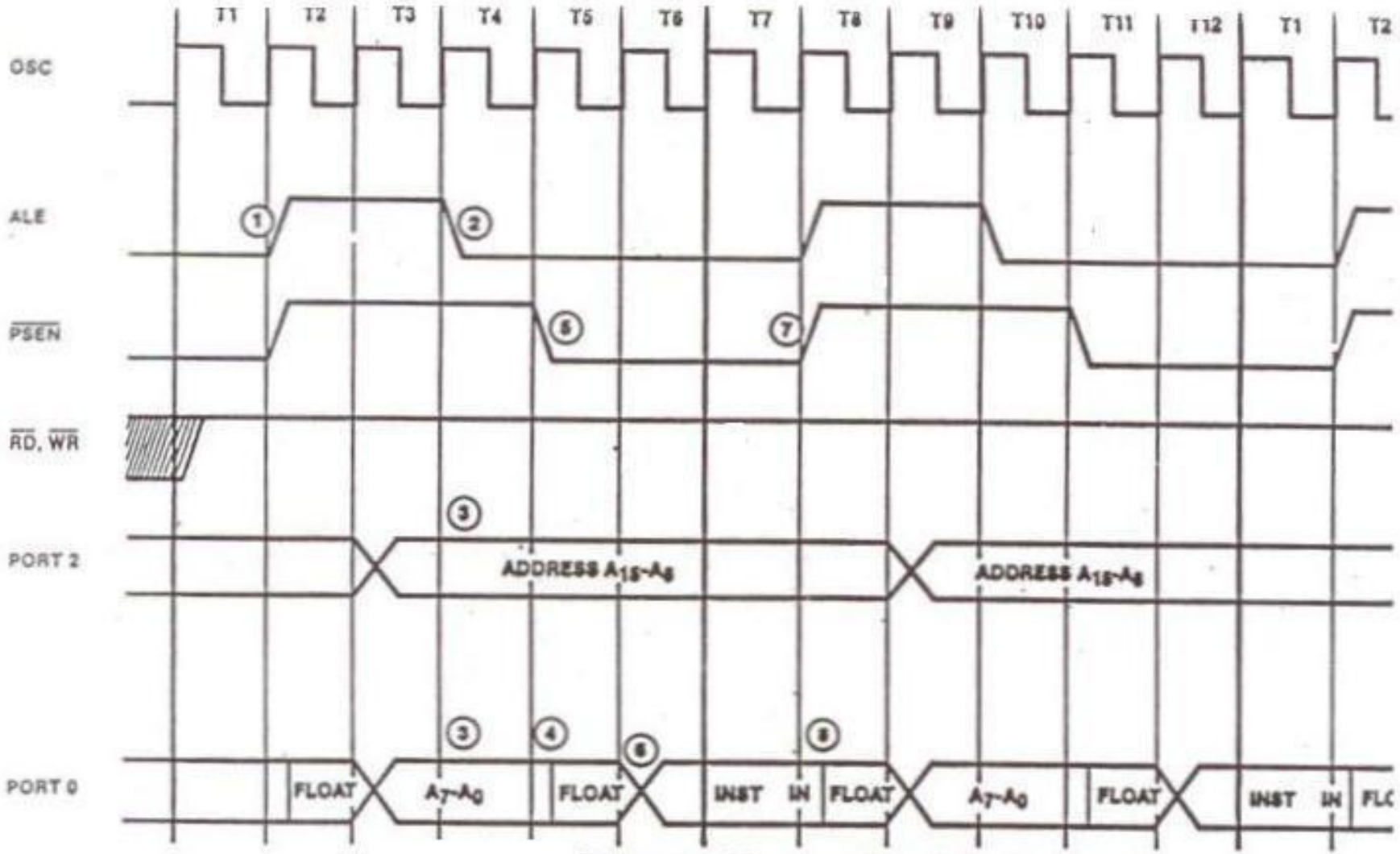

CY
AC
F0
RS1
RS0
OV
—
P
PSW
GATE
C/T̄
M1
M0
GATE
C/T̄
M1
M0
TMOD
TIMER 1
TIMER 0
TF1
TR1
TF0
TR0
IE1
IT1
IE0
IT0
TCON


SM0 SM1 SM2 REN TB8 RB8 TI RI
SCON
EA — — ES ET1 EX1 ET0 EX0
IE
— — — PS PT1 PX1 PT0 PXJ
IP
SMOD — — — — — PD IDL
PCON

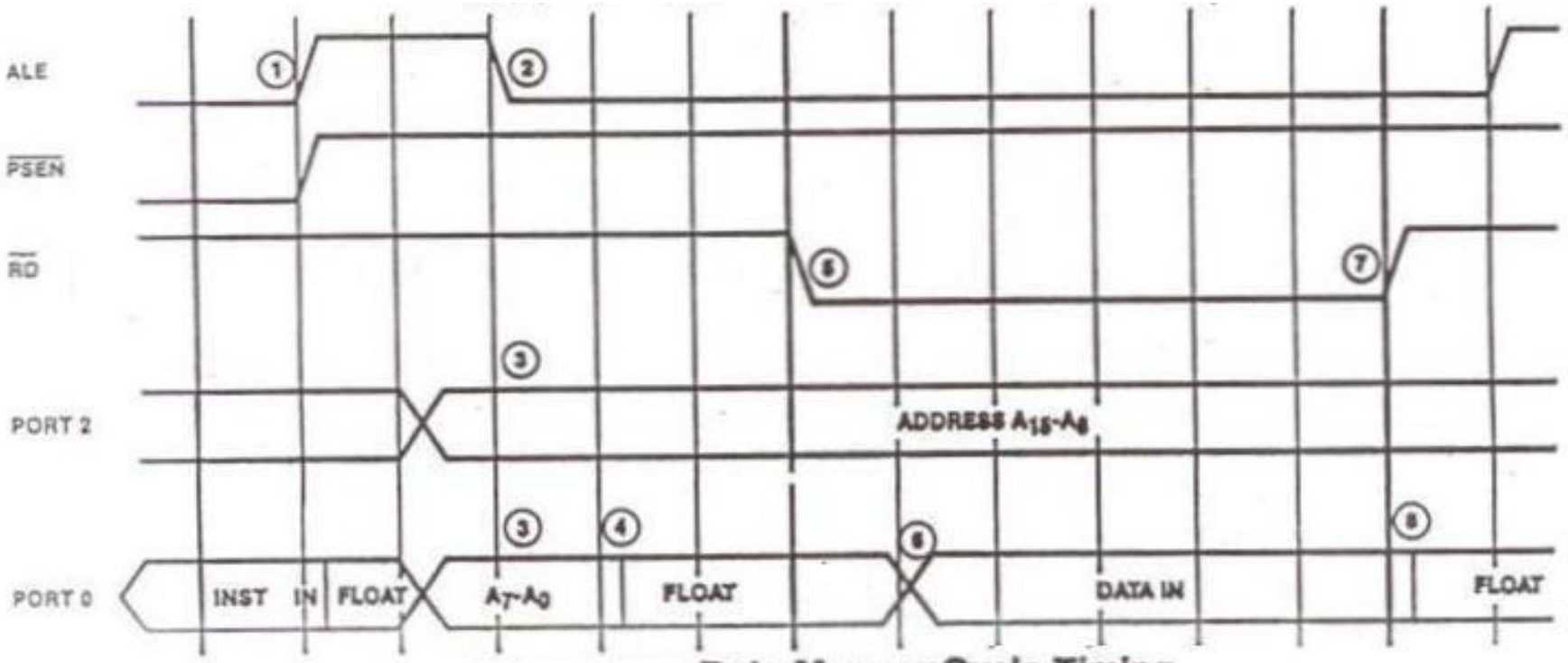
Interrupt Source	Starting Address
External Request 0	3 (0003 H)
Internal Timer/ Counter 0	11 (000B H)
External Request 1	19 (0013 H)
Internal Timer/ Counter 1	27 (001B H)
Internal Serial Port	35 (0023 H)

INC, DEC, CPL, JBC, CJNE, DJNZ, ANL, ORL, XRL

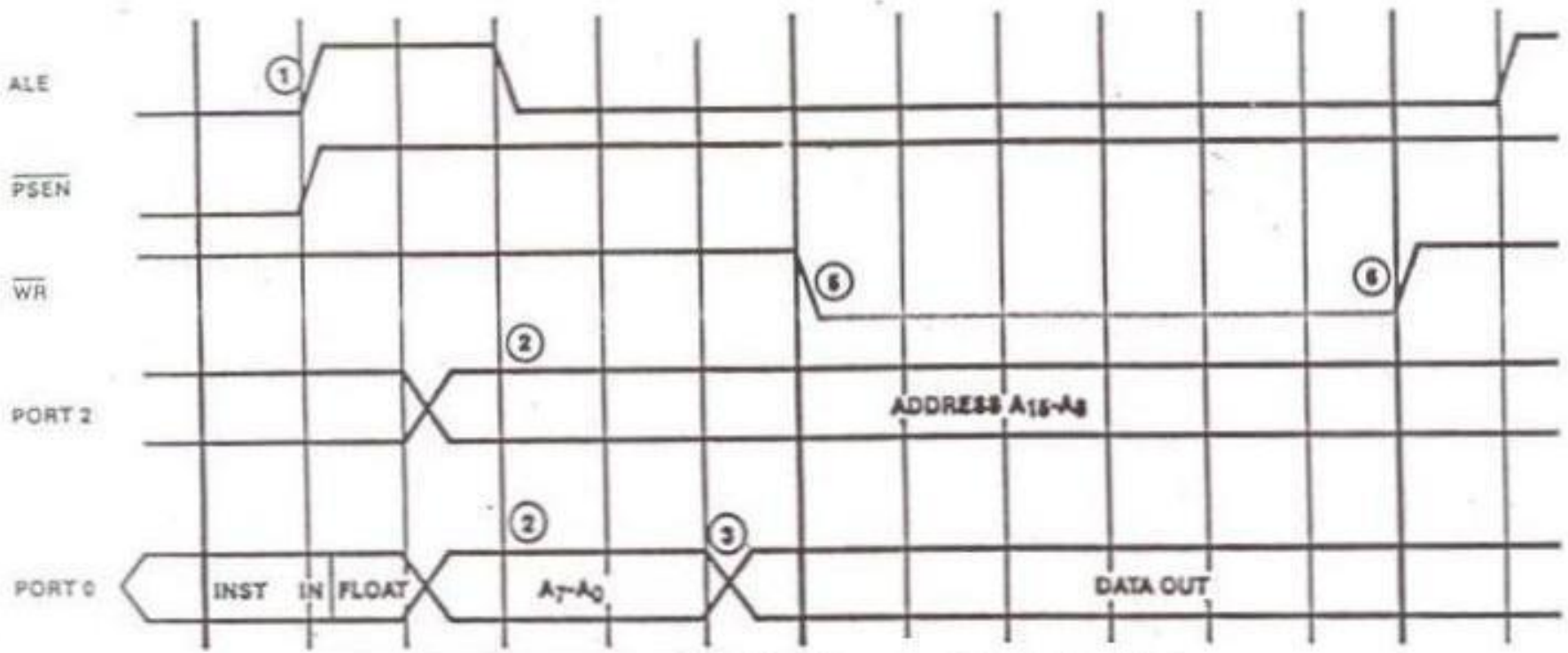




Program Memory Read Cycle Timing



Data Memory Cycle Timing



Data Memory Cycle Timing

Instructions that Affect Flag Settings

Instruction	Flag			Instruction
	C	OV	AC	
ADD	X	X	X	CLR C
ADDC	X	X	X	CPL C
SUBB	X	X	X	ANL C,bit
MUL	O	X		ANL C,/bit
DIV	O	X		ORL C,bit
DA	X			ORL C,/bit
RRC	X			MOV C,bit
RLC	X			CJNE
SETB C	1			

The Instruction Set and Addressing Modes

R_n	Register R7-R0 of the currently selected Register Bank.
direct	8-bit internal data location's address. This could be an Internal Data RAM location (0-127) or a S port, control register, status register, etc. (128-255)].
@R_i	8-bit internal data RAM location (0-255) addressed indirectly through register R1or R0.
#data	8-bit constant included in instruction.
#data 16	16-bit constant included in instruction.
addr 16	16-bit destination address. Used by LCALL and LJMP. A branch can be anywhere within the 64K Memory address space.
addr 11	11-bit destination address. Used by ACALL and AJMP. The branch will be within the same 2K by program memory as the first byte of the following instruction.
rel	Signed (two's complement) 8-bit offset byte. Used by SJMP and all conditional jumps. Range is - bytes relative to first byte of the following instruction.
bit	Direct Addressed bit in Internal Data RAM or Special Function Register.

ARITHMETIC OPERATIONS			
ADD	A,R_n	Add register to Accumulator	1
ADD	A,direct	Add direct byte to Accumulator	2
ADD	A,@R_i	Add indirect RAM to Accumulator	1
ADD	A,#data	Add immediate data to Accumulator	2
ADDC SUBB	A,R_n	Add register to Accumulator with Carry	1
ADDC SUBB	A,direct	Add direct byte to Accumulator with Carry	2
ADDC SUBB	A,@R_i	Add indirect RAM to Accumulator with Carry	1
ADDC SUBB	A,#data	Add immediate data to Acc with Carry	2

INC	A	Increment Accumulator	1
INC	R_n	Increment register	1
INC	direct	Increment direct byte	2
INC	@R_i	Increment direct RAM	1
DEC	A	Decrement Accumulator	1
DEC	R_n	Decrement Register	1
DEC	direct	Decrement direct byte	2
DEC	@R_i	Decrement indirect RAM	1
INC	DPTR	Increment Data Pointer	1
MUL	AB	Multiply A & B	1
DIV	AB	Divide A by B	1
DA	A	Decimal Adjust Accumulator	1

LOGICAL OPERATIONS			
ANL ORL XRL	A,R_n	AND Register to Accumulator	1
ANL	A,direct	AND direct byte to Accumulator	2
ANL	A,@R_i	AND indirect RAM to Accumulator	1
ANL	A,#data	AND immediate data to Accumulator	2
ANL	direct,A	AND Accumulator to direct byte	2
ANL	direct,#data	AND immediate data to direct byte	3

CLR	A	Clear Accumulator	1
CPL	A	Complement Accumulator	1
RL	A	Rotate Accumulator Left	1
RLC	A	Rotate Accumulator Left through the Carry	1
RR	A	Rotate Accumulator Right	1
RRC	A	Rotate Accumulator Right through the Carry	1
SWAP	A	Swap nibbles within the Accumulator	1

DATA TRANSFER			
MOV	A,R_n	Move register to Accumulator	1
MOV	A,direct	Move direct byte to Accumulator	2
MOV	A,@R_i	Move indirect RAM to Accumulator	1
MOV	A,#data	Move immediate data to Accumulator	2
MOV	R_n,A	Move Accumulator to register	1
MOV	R_n,direct	Move direct byte to register	2
MOV	R_n,#data	Move immediate data to register	2

MOV	direct,A	Move Accumulator to direct byte	2
MOV	direct,R_n	Move register to direct byte	2
MOV	direct,direct	Move direct byte to direct	3
MOV	direct,@R_i	Move indirect RAM to direct byte	2
MOV	direct,#data	Move immediate data to direct byte	3
MOV	@R_i,A	Move Accumulator to indirect RAM	1
MOV	@R_i,direct	Move direct byte to indirect RAM	2
MOV	@R_i,#data	Move immediate data to indirect RAM	2

MOV	DPTR,#data16	Load Data Pointer with a 16-bit constant	3
MOVC	A,@A+DPTR	Move Code byte relative to DPTR to Acc	1
MOVC	A,@A+PC	Move Code byte relative to PC to Acc	1
MOVX	A,@R_i	Move External RAM (8-bit addr) to Acc	1
MOVX	A,@DPTR	Move Exernal RAM (16-bit addr) to Acc	1
MOVX	@R_i,A	Move Acc to External RAM (8-bit addr)	1

MOVX	@DPTR,A	Move Acc to External RAM (16-bit addr)	1
PUSH	direct	Push direct byte onto stack	2
POP	direct	Pop direct byte from stack	2
XCH	A,R_n	Exchange register with Accumulator	1
XCH	A,direct	Exchange direct byte with Accumulator	2
XCH	A,@R_i	Exchange indirect RAM with Accumulator	1
XCHD	A,@R_i	Exchange low-order Digit indirect RAM with Acc	1

BOOLEAN VARIABLE MANIPULATION			
CLR	C	Clear Carry	1
CLR	bit	Clear direct bit	2
SETB	C	Set Carry	1
SETB	bit	Set direct bit	2
CPL	C	Complement Carry	1
CPL	bit	Complement direct bit	2
ANL	C,bit	AND direct bit to CARRY	2
ANL	C,/bit	AND complement of direct bit to Carry	2

ORL	C,bit	OR direct bit to Carry	2
ORL	C,/bit	OR complement of direct bit to Carry	2
MOV	C,bit	Move direct bit to Carry	2
MOV	bit,C	Move Carry to direct bit	2
JC	rel	Jump if Carry is set	2
JNC	rel	Jump if Carry not set	2
JB	bit,rel	Jump if direct Bit is set	3
JNB	bit,rel	Jump if direct Bit is Not set	3
JBC	bit,rel	Jump if direct Bit is set & clear bit	3

PROGRAM BRANCHING			
ACALL	addr11	Absolute Subroutine Call	2
LCALL	addr16	Long Subroutine Call	3
RET		Return from Subroutine	1
RETI		Return from interrupt	1
AJMP	addr11	Absolute Jump	2
LJMP	addr16	Long Jump	3
SJMP	rel	Short Jump (relative addr)	2
JMP	@A+DPTR	Jump indirect relative to the DPTR	1

JZ	rel	Jump if Accumulator is Zero	2
JNZ	rel	Jump if Accumulator is Not Zero	2
CJNE	A,direct,rel	Compare direct byte to Acc and Jump if Not Equal	3
CJNE	A,#data,rel	Compare immediate to Acc and Jump if Not Equal	3

CJNE	R_n,#data,rel	Compare immediate to register and Jump if Not Equal	3	
CJNE	@R_i,#data,rel	Compare immediate to indirect and Jump if Not Equal	3	
DJNZ	R_n,rel	Decrement register and Jump if Not Zero	2	
DJNZ	direct,rel	Decrement direct byte and Jump if Not Zero	3	
NOP		No Operation	1	