Fundusze Europejskie
Wiedza Edukacja Rozwój

Rzeczpospolita Polska

Unia Europejska
Europejski Fundusz Społeczny

Politechnika Śląska jako Centrum Nowoczesnego Kształcenia
opartego o badania i innowacje

POWR.03.05.00-IP.08-00-PZ1/17

Projekt współfinansowany przez Unię Europejską ze środków Europejskiego Funduszu Społecznego

Microprocessor and Embedded Systems

Faculty of Automatic Control, Electronics and Computer Science, Informatics, Bachelor Degree

Lecture 11

PIC-family single-chip microcomputers
Part 2
Built-in peripherials

Bartłomiej Zieliński, PhD, DSc

PIC (2)

Program:

(last week)

- Von Neumann vs Harvard architecture
- PIC structure & memory organisation

(today)

- I/O ports
- Timers/counters
- Interrupts

PIC (2)

- PIC – clock
  - Clock modes

FOSC	Mode	Descripion	f_{max}	Remarks
11	RC	RC generator	8/20 MHz	Low cost, low stability
10	HS	Quartz generator, high freq.	8/20 MHz	High speed, high power
01	XT	Standard quartz generator	4 MHz	Std. freq., average power
00	LP	Quartz oscillator	500 kHz	Low frequency, low power

- Modes HS, XT, LP: external TTL generator possible
  - OSC1 = input
  - OSC2 – unused
- Mode RC:
  - OSC1 = input
  - $f_{OSC2} = ¼\ f_{OSC1}$

PIC (2)

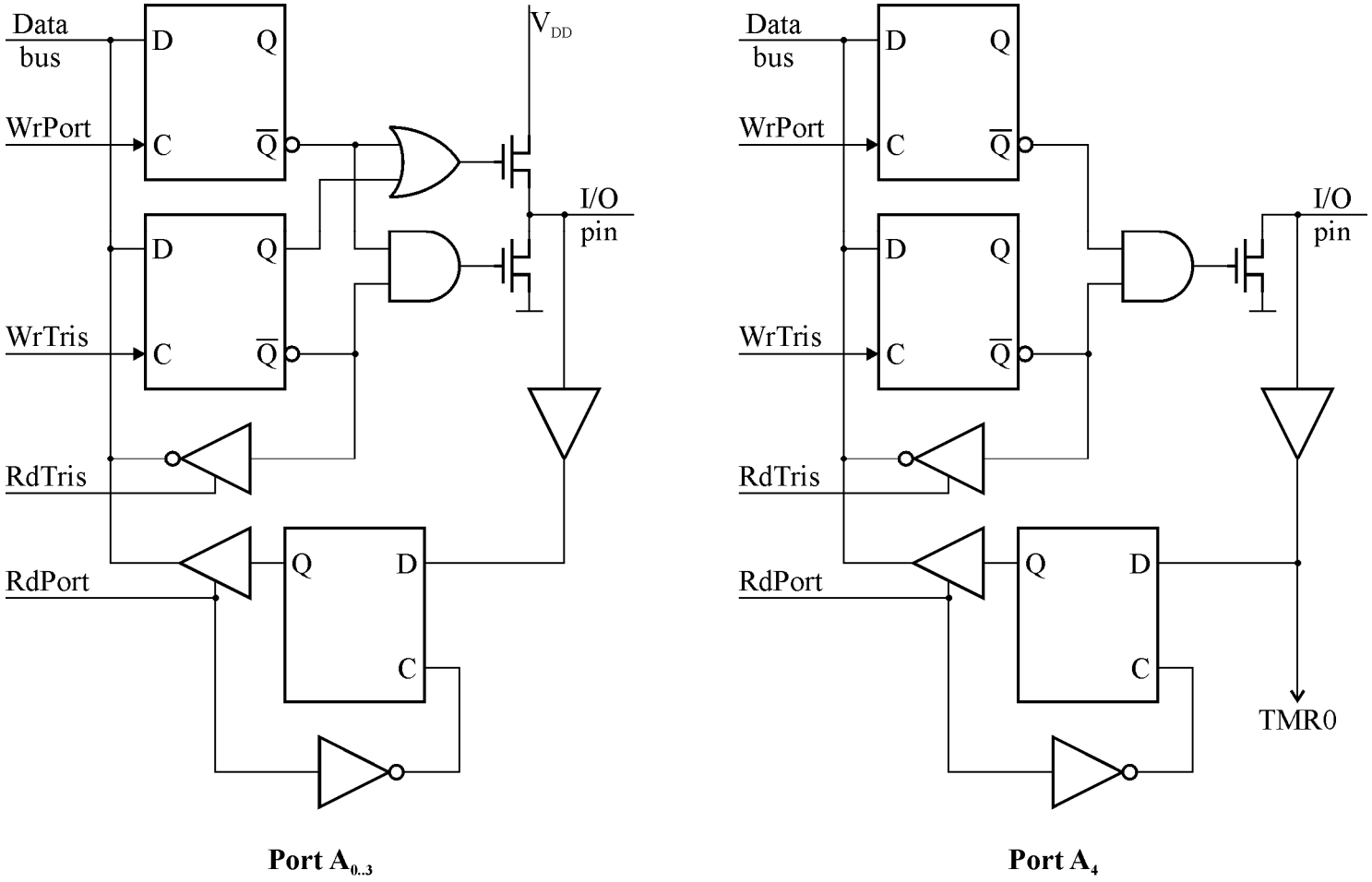
- PIC – reset
  - Reset reason identification
    - Power-on reset,
    - !MCLR during normal operation
    - !MLCR during sleep
    - Watchdog during normal operation
    - Watchdog during sleep
  - Depending on reason, various registers are initialised
  - After reset: Q1 phase, PC=0000h
  - PWRT (*Power-on timer*): 72 ms
  - OST (*Oscillator Start-up Timer*): 1024 clocks

PIC (2)

- PIC – I/O ports
  - Number of ports & lines depends on µc type
  - Port → 2 registers
    - PORT – for data I/O
      - Write → port register
      - Read → pin state
    - TRIS – direction control (1nput, 0utput)
  - Analogue inputs
    - After reset – analogue input
    - Analogue off, TRIS set → digital

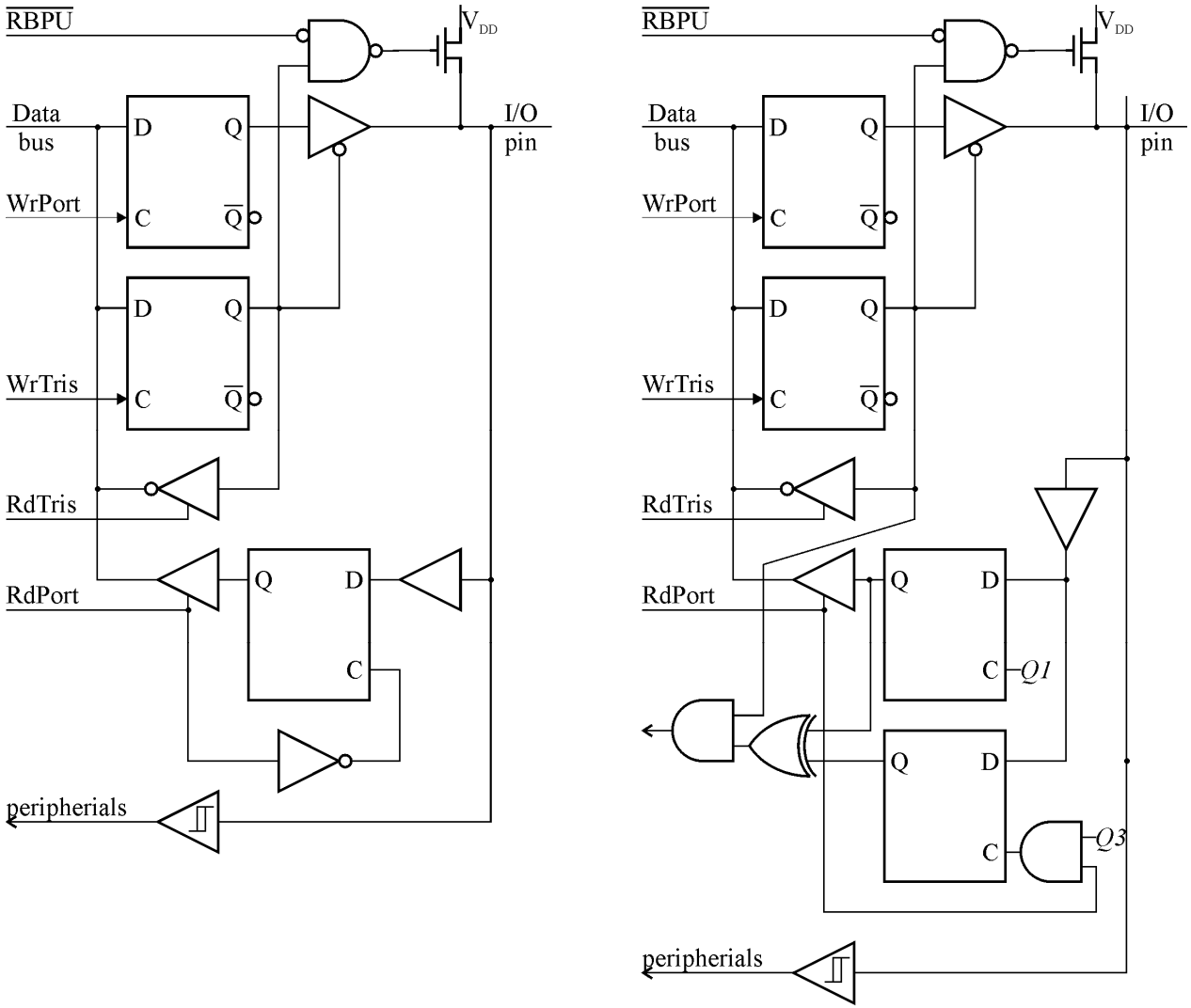
PIC (2)

- PIC – Port A



PIC (2)

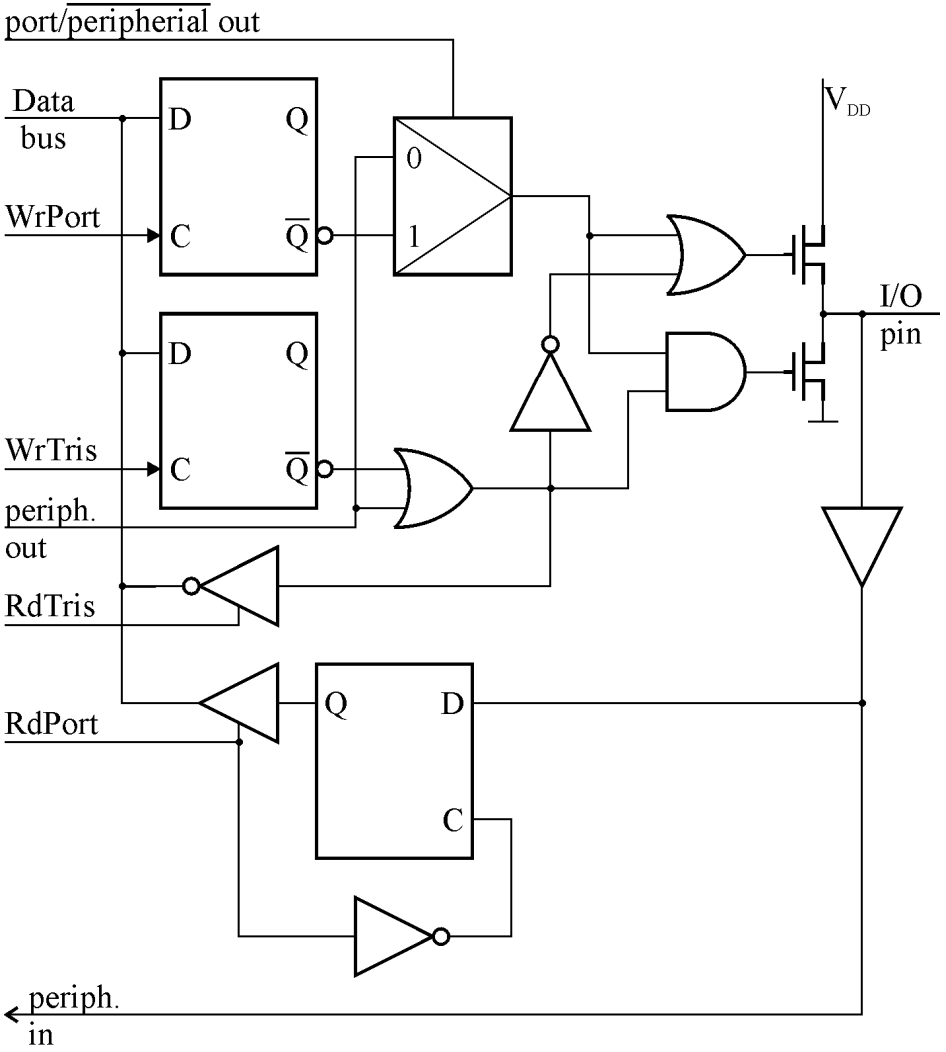
- PIC – Port B



Port $B_{0..3}$ **Port $B_{4..7}$**

PIC (2)

- PIC – Port C



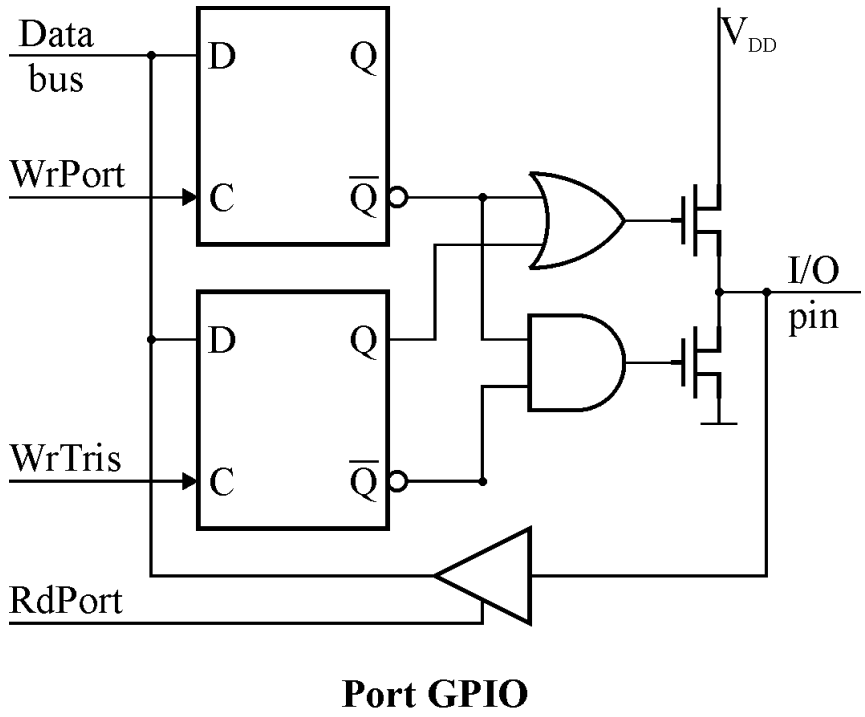
Port C

PIC (2)

- PIC – Ports D, E
  - Structure like Port A
  - May work as PSP (*Parallel Slave Port*)
    - Port D (8-bit) = Data
    - Port E (3-bit) = $\overline{\mathrm{RD}}$, $\overline{\mathrm{WR}}$, $\overline{\mathrm{CS}}$
    - Can be directly connected to a µp's bus
- Port read-after-write
  - Effective write – Q4 phase
  - `nop` recommended before read

PIC (2)

- PIC – Port GPIO
  - Only in 8-pin PIC's
    - E.g., PIC12C5xx
  - 5 I/O lines, 1 input line
  - Can't read TRIS
  - RdPort=RdPin
  - PIC12C67x
    - Like PortA, PortB



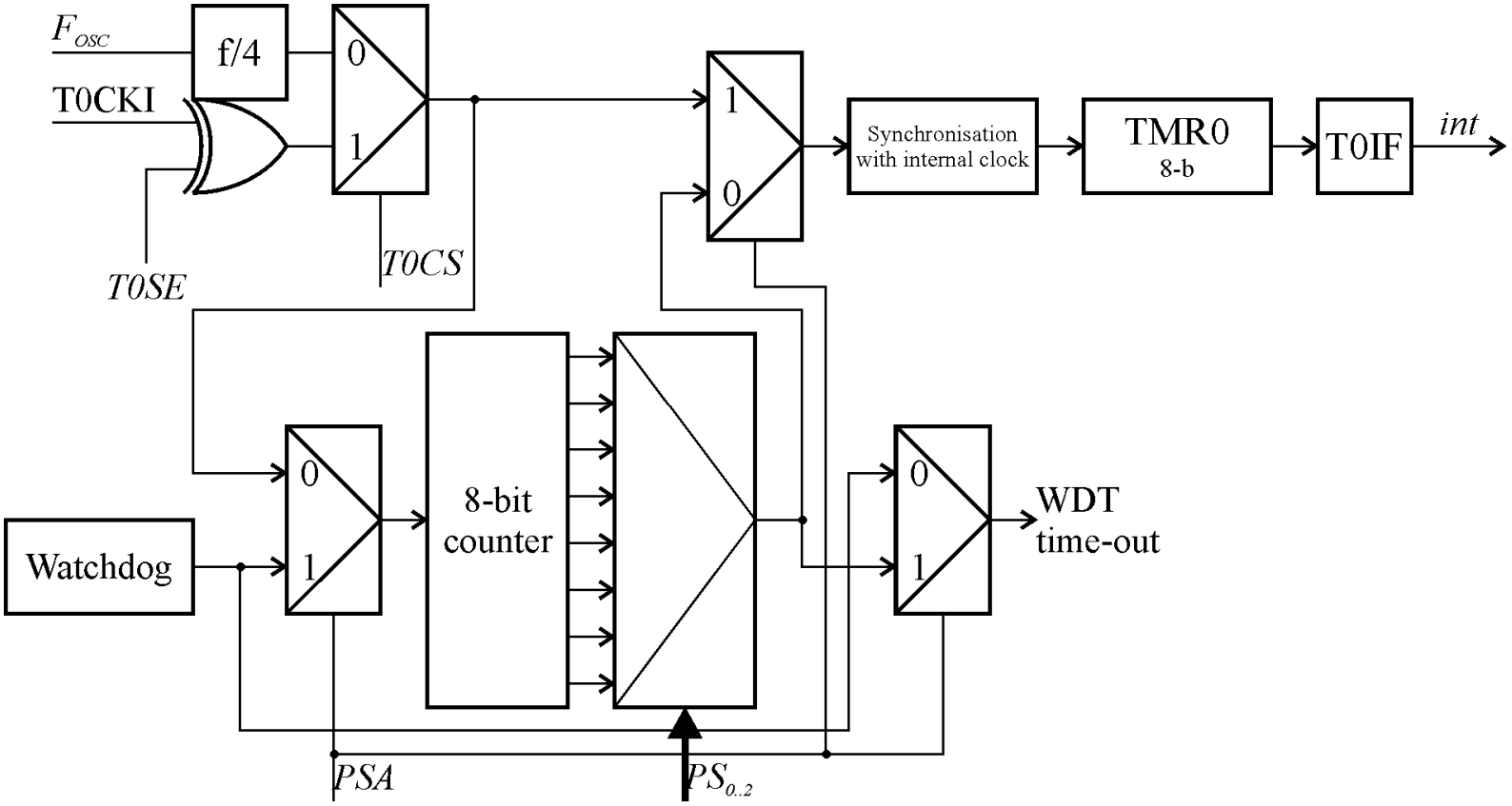
Port GPIO

PIC (2)

- PIC – Timer0
  - Timer (T0CS=0)/counter (T0CS=1) mode
  - Programmable T0CKI active edge
  - TMR0
    - Counts down
    - Read any time
    - Write → count pause for 2 T_{clk}
  - Prescaler: 1:1…1:256
    - PSA=1 or 1:1 – prescaler works for watchdog

PIC (2)

- PIC – Timer0



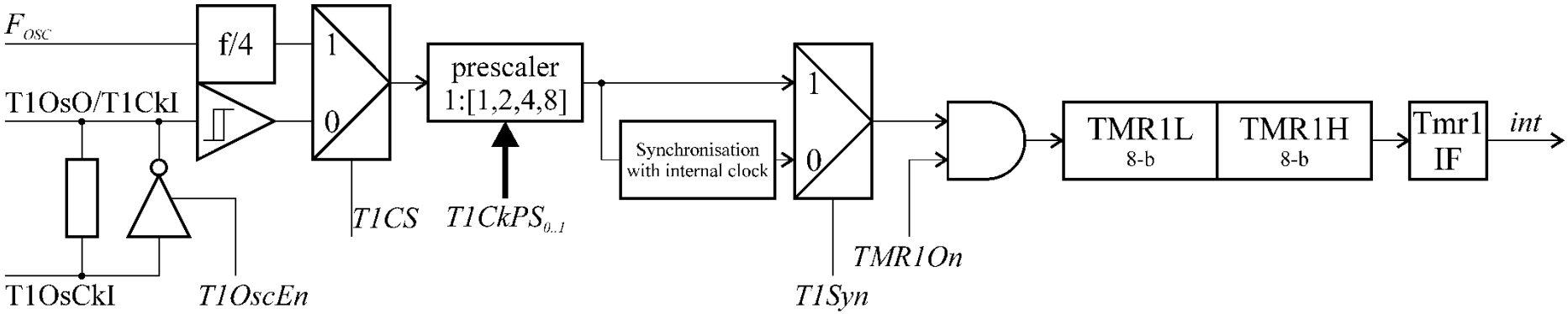
  - OPTION_REG
    - T0SE, T0CS, PSA, $PS_{0..2}$
  - INTCON
    - T0IF

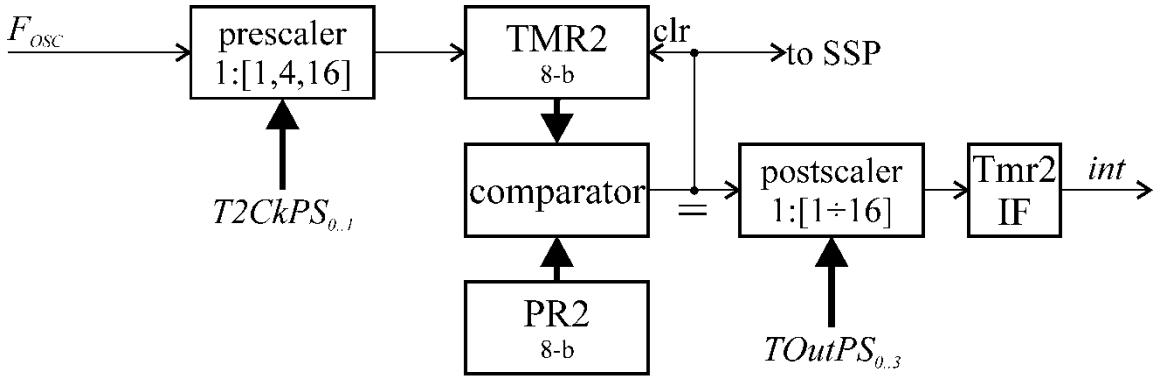
PIC (2)

- PIC – Timer 1
  - Registers:
    - TMR1H, TMR1L – 2×8 bits
    - T1CON – control bits
    - PIR1.TMR1F – interrupt flag
    - PIE1.TMR1E – interrupt enable
  - Input:
    - Internal (1/4 f_{osc})
    - External TTL
    - External quartz resonator (32…200 kHz)

PIC (2)

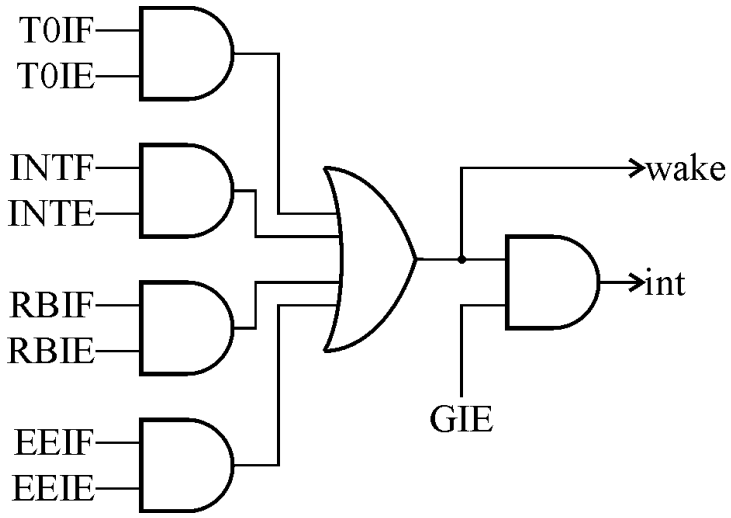
- PIC – Timers 1 & 2





PIC (2)

- PIC – interrupts
  - 1-level system, no priority
  - Typically 1 int/module
  - Registers
    - INTCON – basic
      - GIE – Global Interrupt Enable
      - Individual:
        - » RBIE – Port B change,
        - » T0IE – Timer 0,
        - » INTE – external int.



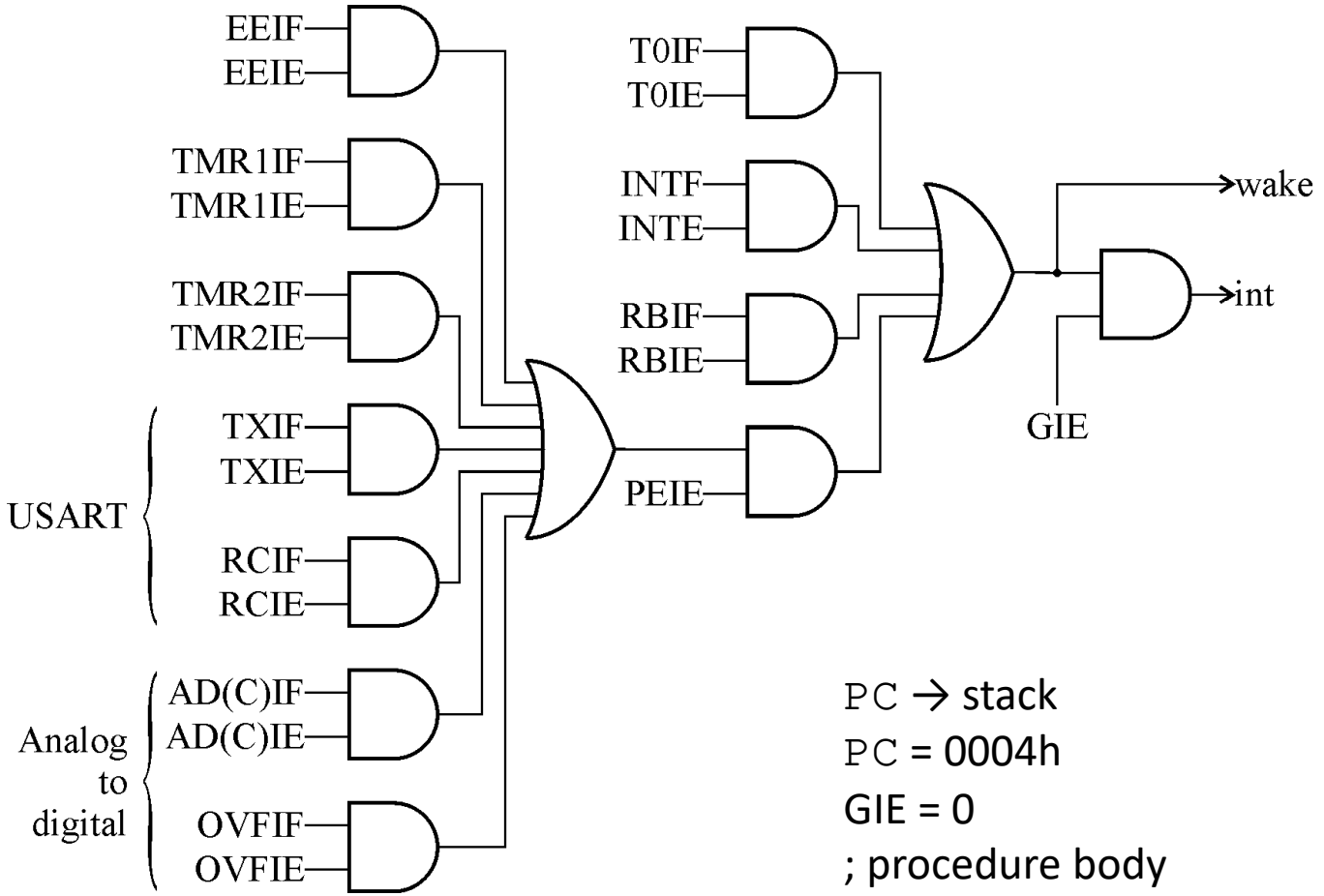
    - PIE1, PIR1, PIE2, PIR2, etc. – more complex PIC's

PIC (2)

- PIC – interrupts
  - Sources:
    - External interrupt
    - Timer 0 overflow
    - Port B change
    - Comparator change
    - Parallel Slave Port
    - USART
    - A/D conversion complete
    - Data EEPROM write complete
    - Timer 1 overflow
    - Timer 2 overflow
    - CCP, SSP, etc.

PIC (2)

- PIC – interrupts



`PC` → stack
`PC` = 0004h
GIE = 0
; procedure body
`retfie` ; GIE=1, stack → `PC`

PIC (2)

- PIC – interrupts
  - Interrupt accepted, if active and unmasked
    - Masked interrupt flag also set
      - Can be checked after unmasked (time-critical code fragments)
      - Must clear flags before unmasking
      - GIE=0 doesn't clear flags → acceptance after GIE=1
    - Acceptance delay
      - Interrupt identification, service preparation
      - Internal int's – 3 command cycles
      - External int's – 4 command cycles (not synced with µp)
    - Register content
      - Saved in GPR (no sofware accessible stack)
      - Must not change flags, e.g., STATUS

PIC (2)

- PIC – communication
  - Serial interfaces
    - USART
    - SPI
    - CAN
    - USB
    - LIN
    - SSP – I^2C, SPI
  - Parallel interfaces
    - PSP

PIC (2)

- PIC – USART
  - Modes
    - Asynchronous, full-duplex
    - Synchronous, half-duplex, master
    - Synchronous, half-duplex, slave
  - Registers
    - SPBRG – baud rate generator
    - RxSTA – receiver status
    - TxSTA – transmitter status
    - TxREG – transmitter data register
    - RxREG – receiver data register

PIC (2)

- PIC – USART
  - TxSTA
    - CSRC – clock source for synchronous transmission
    - TX9 – 8/9-bit word
    - TxEN – transmitter enable
    - SYNC – synchronous/asynchronous transmission
    - BRGH – slow/fast transmission
    - TRMT – transmitter shift register full/empty
    - Tx9D – 9^{th} data bit (transmitter)

PIC (2)

- PIC – USART
  - RxSTA
    - SPEN – USART enable/disable
    - RX9 – 8/9 data bits
    - SREN – syn transm: single frame reception
    - CREN – continuous reception
    - ADDEN – address detection in 9-bit mode
    - FERR – frame reception error
    - OERR – overflow error
    - RX9D – 9^{th} data bit (receiver)

PIC (2)

- PIC – USART
  - SPBRG – baud rate generation
    - 8-bit register
    - Asynchronous transmission
      - Fast mode: $R = \frac{f_{clk}}{16(\text{SPBRG}+1)}$
      - Slow mode: $R = \frac{f_{clk}}{64(\text{SPBRG}+1)}$
    - Synchronous transmission
      - Fast mode only: $R = \frac{f_{clk}}{4(\text{SPBRG}+1)}$

PIC (2)

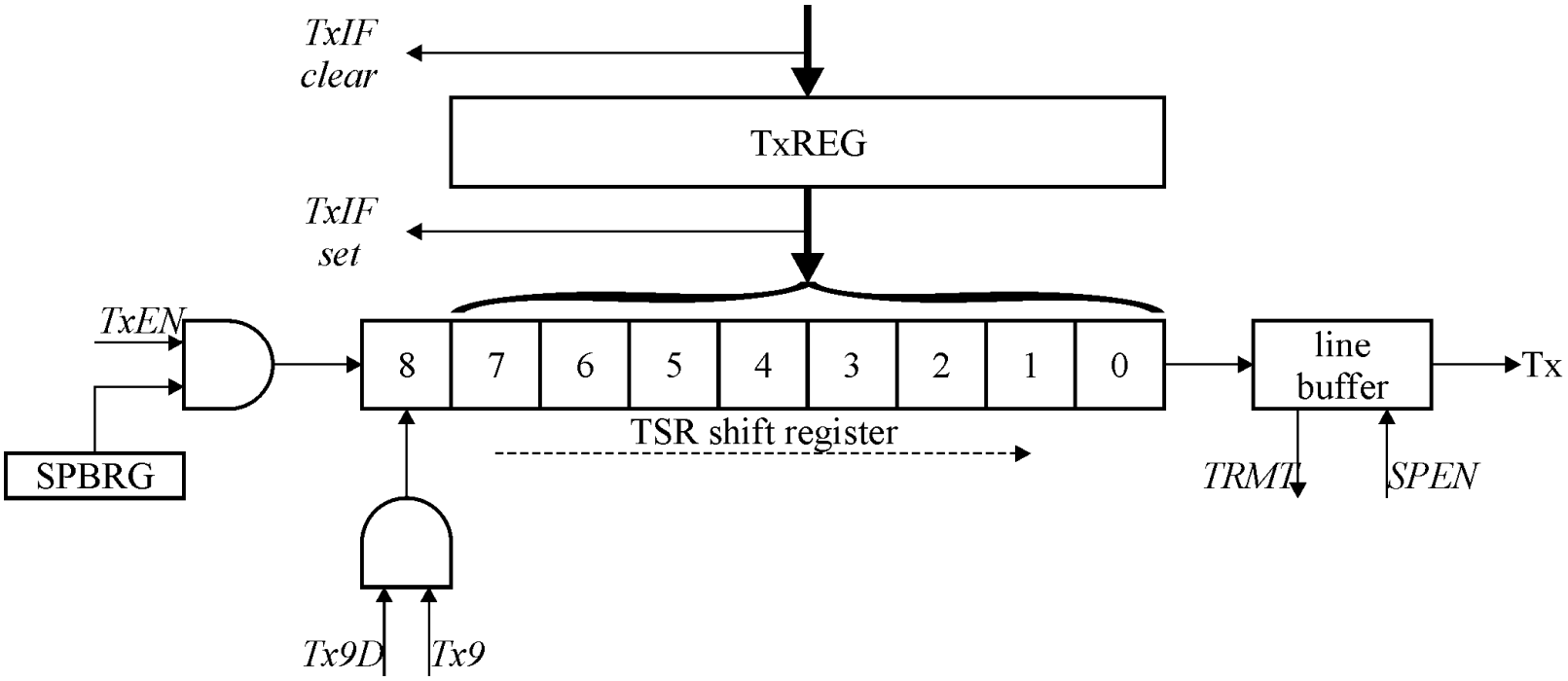
- PIC – USART
  - Baud rate generation precision
    - Asynchronous transmission

	Slow mode			Fast mode		
Rate	X	X’	Δ[%]	X	X’	Δ[%]
2.4	25.04	25	0.15	103.16	103	0.16
4.8	12.02	12	0.16	51.08	51	0.16
9.6	5.51	5-6	10.2	26.04	26	0.16
14.4	3.34	3	11.3	16.3	16	2.2
19.2	2.26	2	11.3	12.02	12	0.16

      - Higher transmission rates → fast mode recommended
        - » 20 MHz µc necessary
        - » Higher power consumption

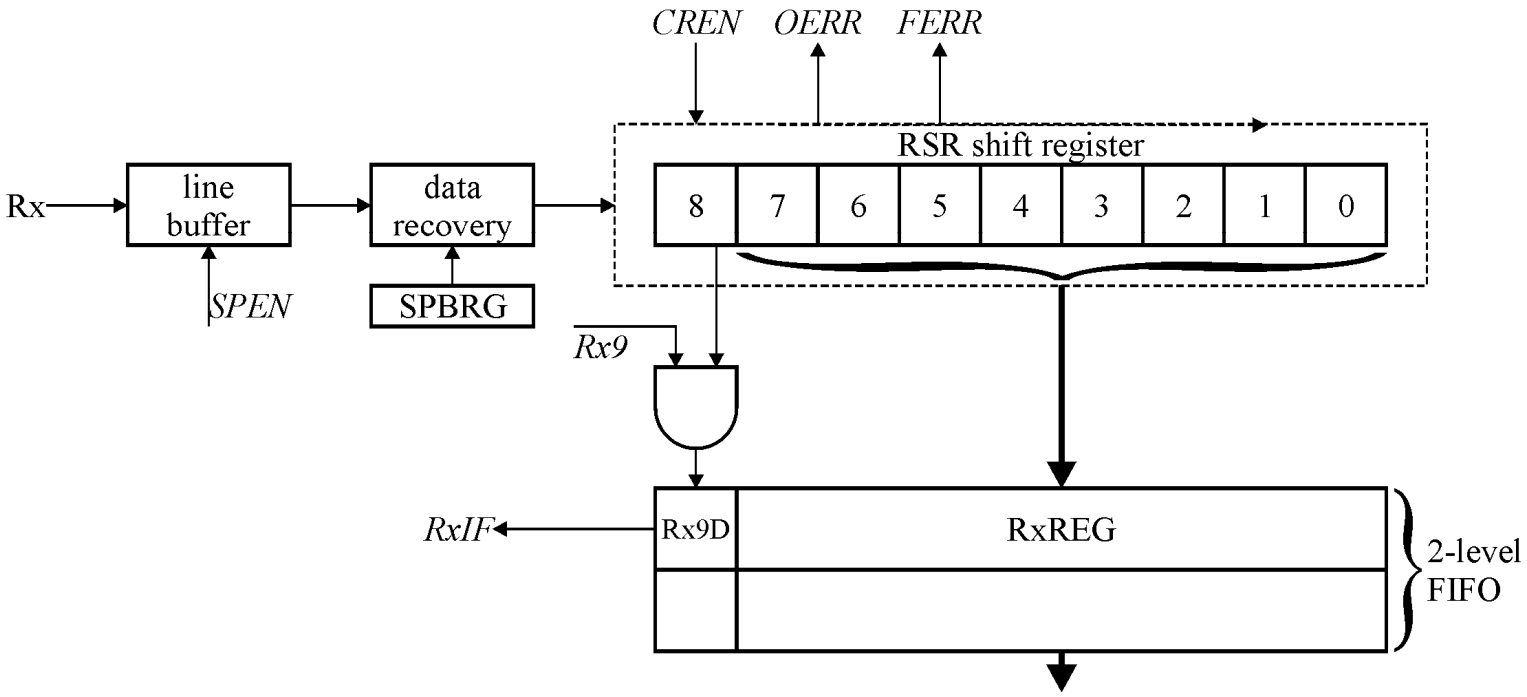
PIC (2)

- PIC – USART
  - Transmitter structure (async. transm.)



PIC (2)

- PIC – USART
  - Receiver structure (async. transm.)

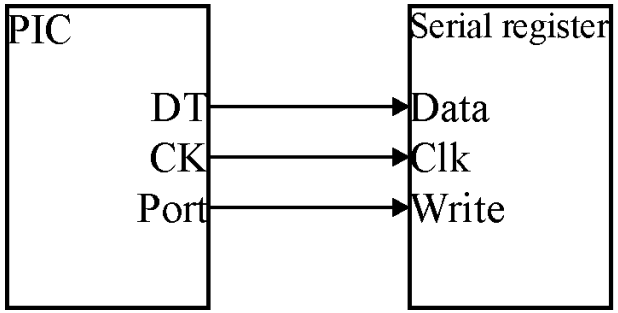


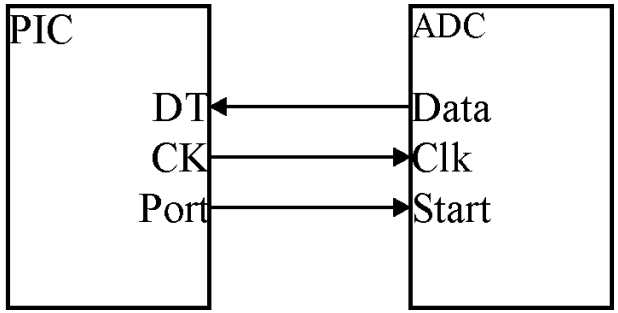
PIC (2)

- PIC – USART
  - Multiprocessor communication
    - RxSTA.ADDEN = 1
      - Master: send address (Rx9D=1)
      - Selected slave: ADDEN=0
      - Master: send data (Rx9D=0)
    - Similar to 8051 special mask mode

PIC (2)

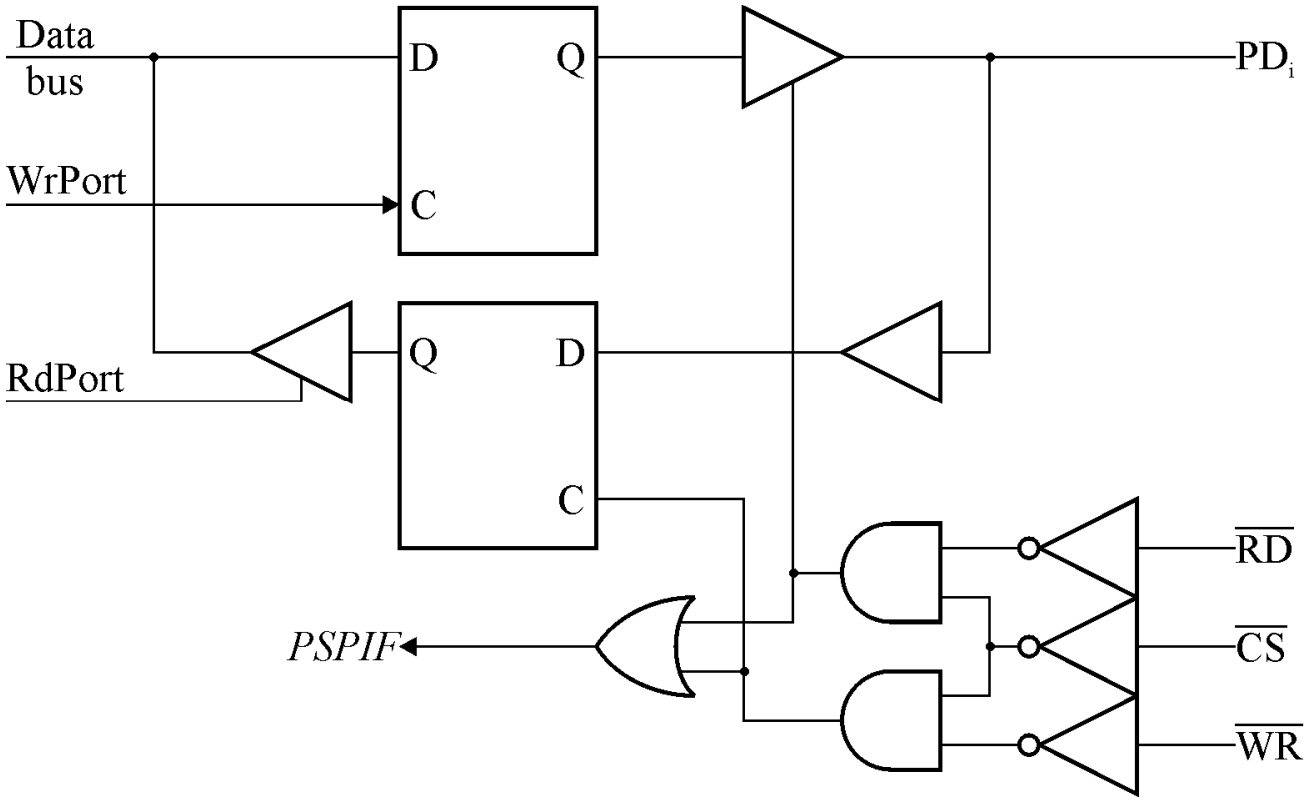
- PIC – USART
  - Synchronous transmission
    - Write, e.g. serial-input register
    - Read, e.g. serial-output AD converter
      - E.g., MCP3201





PIC (2)

- PIC – PSP
  - Can be connected directly to a master system bus
  - PIC as peripherial (slave) µp



PIC (2)

- PIC – PSP
  - TRISE register (PortE – 3 bits only)
    - IBF – 1=input buffer full
    - OBF – 1=output buffer full
    - IBOV – 1=input buffer overflow
      - must be software cleared
    - PSPMODE – 1=on
    - B2, B1, B0 – line I/O
  - PortE configuration
    - Set as digital (default=analogue)
      - $ADCON1.PCFG_{0..3}$
    - Set as inputs
      - $TRISE.B_{0..2}$

PIC (2)

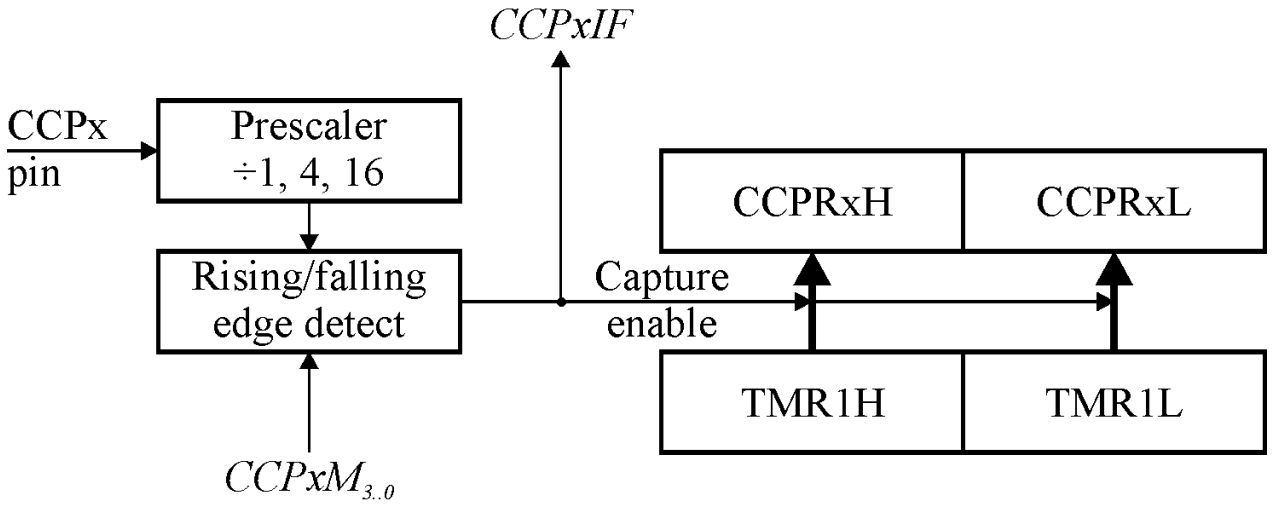
- PIC – CCP
  - Compare, Capture, PWM
  - 1-2 independent modules per µc, but
    - Common f_{out}
    - Capture mode → both use TMR1
    - Compare mode → config with TMR1 clear must be selected
  - Consists of:
    - CCPRxH, CCPRxL registers (x=0, 1)
    - 16-b comparator

PIC (2)

- PIC – CCP
  - CCPxCON register
    - $DCxB_{1..0}$ – (PWM mode) PWM duty cycle Lsbits
      - Upper 8 bits in CCPRxL
    - $CCPxM_{3..0}$ – mode:
      - CCP off
      - Capture mode
        - » every falling edge
        - » every rising edge
        - » every 4^{th} rising edge
        - » every 16^{th} rising edge
      - Compare mode
        - » Initialize CCP pin Low, on compare match force CCP pin High
        - » Initialize CCP pin High, on compare match force CCP pin Low
        - » Generate software interrupt on compare match
        - » Trigger special event
      - PWM mode

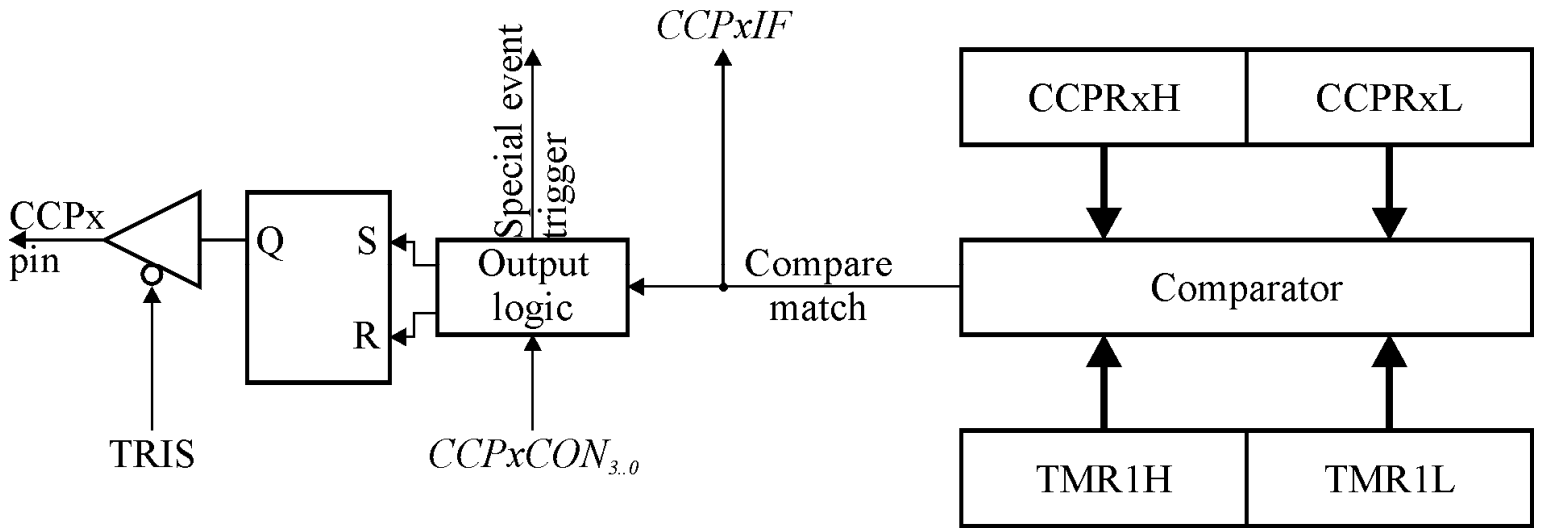
PIC (2)

- PIC – CCP
  - Capture mode
    - Capture TMR1 value on event occurence
      - CCPx pin as input (if output, accidental capture may occur)
    - TMR1 not reset
      - TMR1 usable for other operations
      - TMR1: timer mode or synchronised counter mode



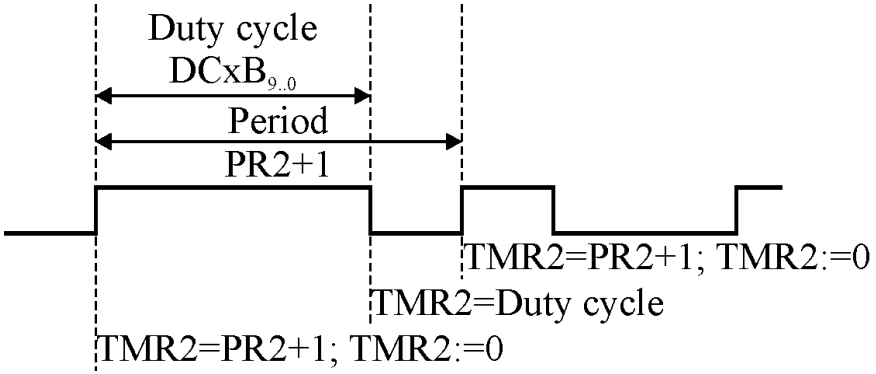
PIC (2)

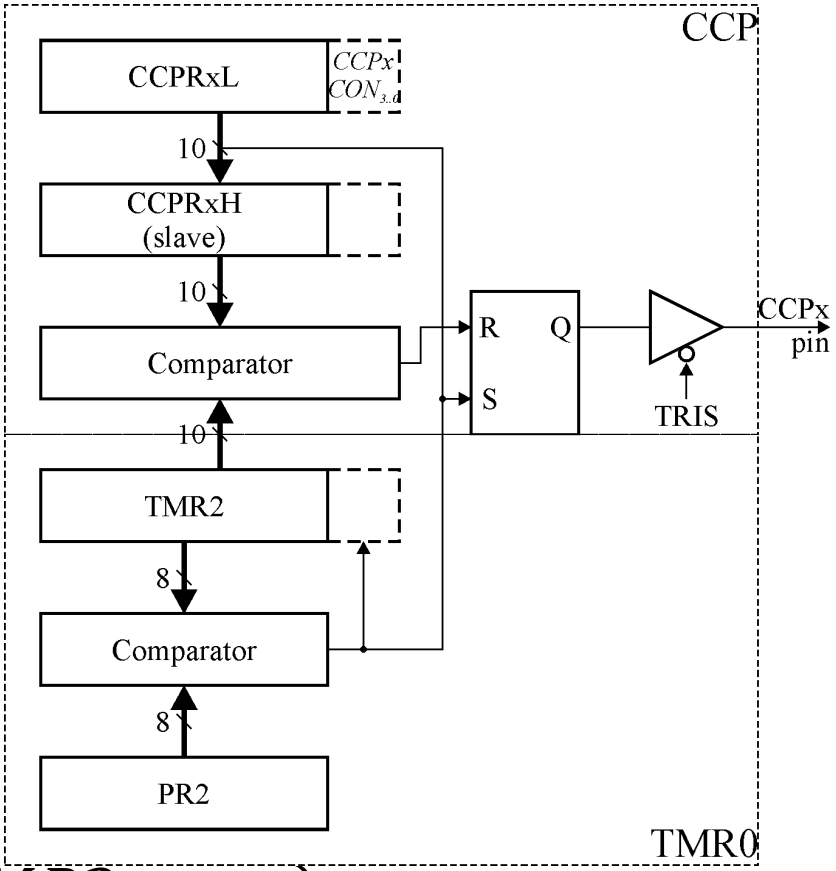
- PIC – CCP
  - Compare mode
    - On compare match:
      - CCPx output may change (high/low/no change)
      - Interrupt generated
    - TMR1: timer mode or synchronised counter mode



PIC (2)

- PIC – CCP
  - PWM mode

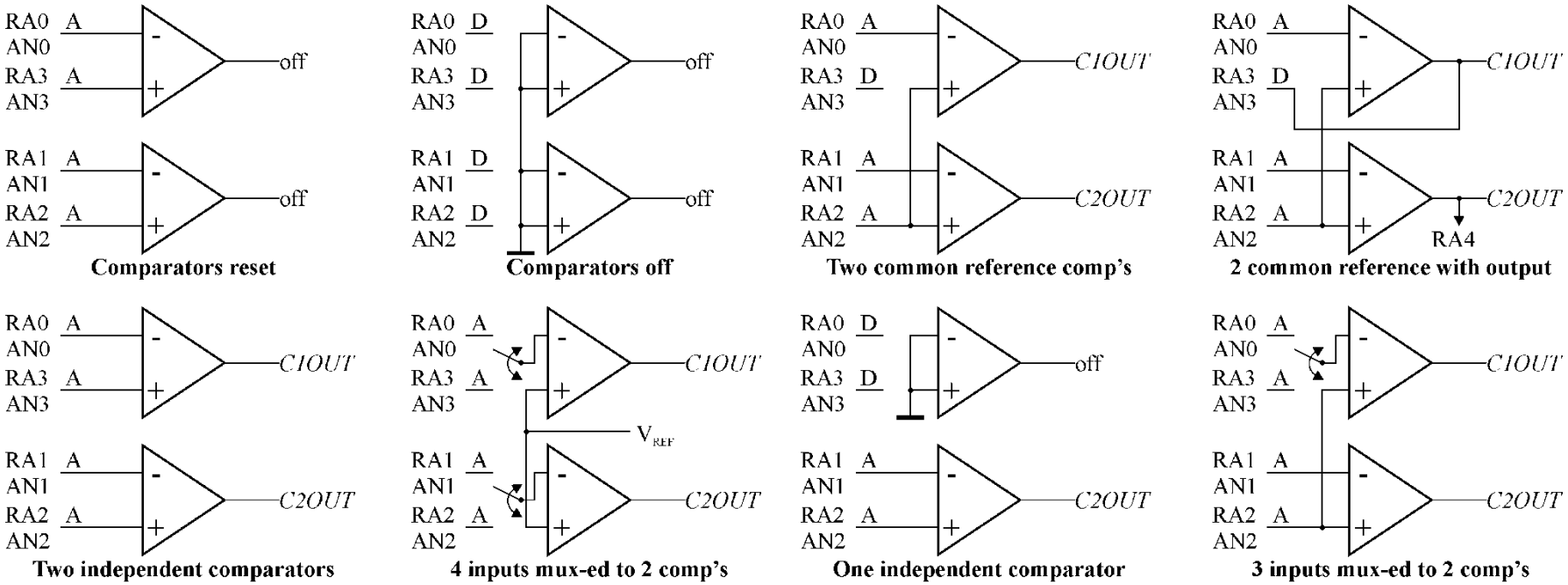




- $T_{PWM} = 4T_{clk}(PR2 + 1)(TMR2presc)$
- $\eta_{PWM} = T_{clk}DCxB_{9..0}\ (TMR2presc)$
- $\eta_{PWM} > T_{PWM} \rightarrow$ CCPx=1; $\eta_{PWM} = 100\%$

PIC (2)

- PIC – analogue comparators
  - 8 modes



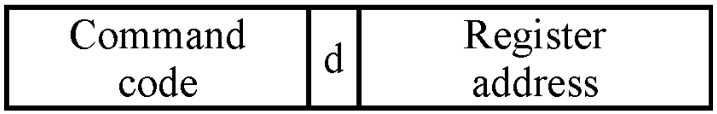
PIC (2)

- PIC – analogue comparators
  - CMCON register
    - C1OUT, C2OUT (comparators outputs)
    - C1INV, C2INV (direct/complemented outputs)
    - CIS – input multiplexer
    - $CM_{2..0}$ – mode
  - Configuration
    - CMCON set
    - I/O lines configuration
    - CMIF=0, CMIE=1 (if interrupts used)
    - V_{REF} configuration (if used)

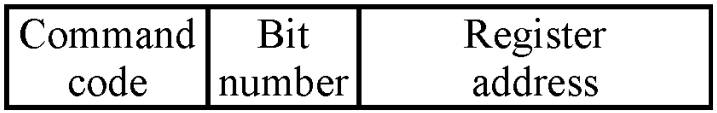
PIC (2)

- PIC – instruction set
  - Instruction formats

Byte commands



Bit commands



Control & constant commands

Jump & call

- Some instructions → no fields defined

PIC (2)

- PIC – instruction set
  - Data transfer
    - Movlw k; k→W
    - Movwf f; W→[f]
    - Movf f,d; [f]→W (d=0), [f]→[f] (d=1)
  - Addition
    - Addlw k; k+W→W
    - Addwf f,d; W+[f]→W (d=0), →[f] (d=1)
  - Substraction
    - sublw k; k-W→W
    - subwf f,d; [f]-W→W (d=0), →[f] (d=1)

PIC (2)

- PIC – instruction set
  - Inc/dec
    - Incf f,d; [f]+1→W (d=0), →[f] (d=1)
    - decf f,d; [f]-1→W (d=0), →[f] (d=1)
  - Logical
    - Andlw k; W AND k →W
    - Andwf f,d; [f] AND W→W (d=0), →[f] (d=1)
    - Iorlw k; W OR k →W
    - Iorwf f,d; [f] OR W→W (d=0), →[f] (d=1)
    - Xorlw k; W XOR k →W
    - Xorwf f,d; [f] XOR W→W (d=0), →[f] (d=1)
    - comf; NOT [f] →W (d=0), →[f] (d=1)

PIC (2)

- PIC – instruction set
  - Rotation (with carry flag)
    - rlf f,d; [f]<<1→W (d=0), →[f] (d=1)
    - rrf f,d; [f]>>1→W (d=0), →[f] (d=1)
  - Swap
    - Swapf f,d
  - Bit set/reset
    - Bsf f,b
    - Bcf f,b

PIC (2)

- PIC – instruction set
  - Jumps & calls
    - Goto s; s→PC, PCLATH→PCH
      - Goto $; Infinite loop
    - Call s
    - Return
    - Retlw k; return with W=k
      - Can be used e.g. in computed goto
    - <instr> pcl, f; computed goto
      - Addwf pcl, f; pcl+w→pcl
      - Movwf pcl; w→pcl

PIC (2)

- PIC – instruction set
  - Conditional jumps
    - *BaseLine & Midrange: if condition fulfiled, skip next instr.*
    - Btfss f,b; jump if bit set
    - Btfsc f,b; jump if bit cleared
    - Incfsz f,d; inc & skip if $result_{0..7}=0$
    - Decfsz f,d; dec & skip if $result_{0..7}=0$

PIC (2)

- PIC – instruction set
  - Other instructions
    - Retfie; return from interrupt: return + GIE=1
    - Sleep; goto low power mode
    - Clrwdt; watchdog timer clear
    - Nop
    - Tris n; W→ TRIS
      - Obsolete; →movwf trisx
    - Option; W→ OPTION_REG
      - Obsolete; →movwf option_reg