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 1
General architecture

Bartłomiej Zieliński, PhD, DSc

PIC (1)

Program:

(today)

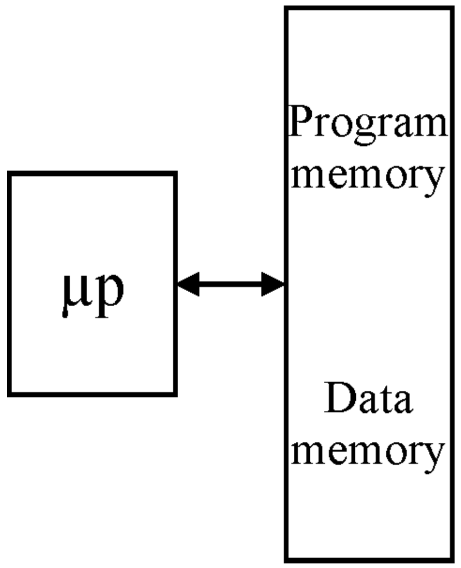
- Von Neumann vs Harvard architecture
- PIC structure
- Memory organisation

(next week)

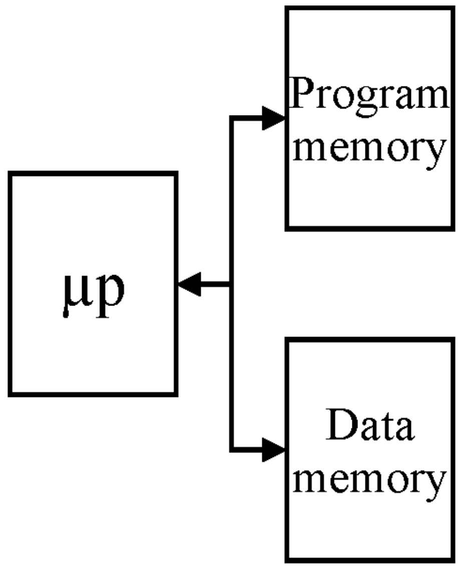
- Built-in peripherials

PIC (1)

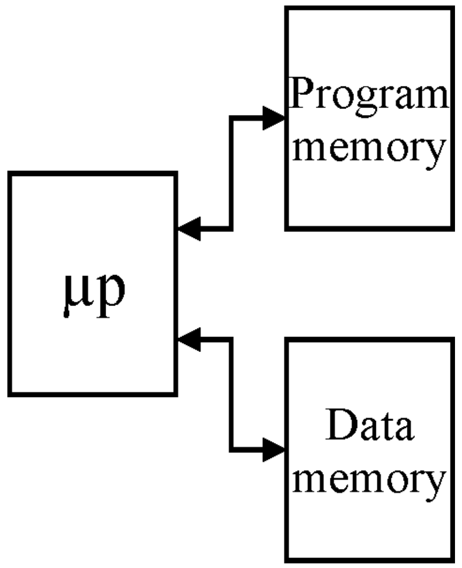
- Von Neumann vs Harvard architecture



von Neumann architecture



"Intermediate" architecture



Harvard architecture

PIC (1)

- Von Neumann vs Harvard architecture
  - Separate buses for program and data memories
    - Data memory word length ≠ program memory word length
      - Command code = 1 program memory word
      - All code and data present in a single word
      - Command code length and format can differ accross µp versions
        - Source-code compatibility
    - Independent program and data memories access
      - Parallel access
        - E.g., command 1 writes while command 2 is being fetched
        - Faster program execution → pipellining
    - Address ambiguity
      - Must also specifiy program/data memory space

PIC (1)

- PIC families

Family	Program	Commands	Examples
Base-line	12 b	33	PIC12Cxxx, PIC16Cx
Mid-Range	14 b	35	PIC16C/Fxxx, PIC12C6xx
High End	16 b	55	PIC17xxx
Enhanced	16 b	77	PIC18xxx
dsPIC			

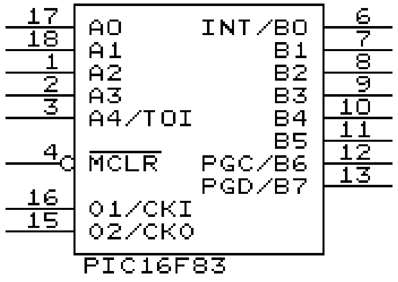
- Versions
  - F = Flash
  - C = OTP EPROM
  - CR = ROM
  - CE = OTP + EEPROM
  - HV = High voltage (15V)
  - L = Low voltage (2÷5.5V)
  - E.g., LF, LCR, etc.

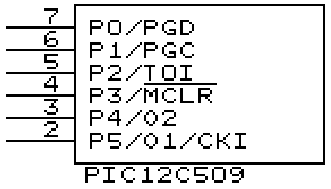
PIC (1)

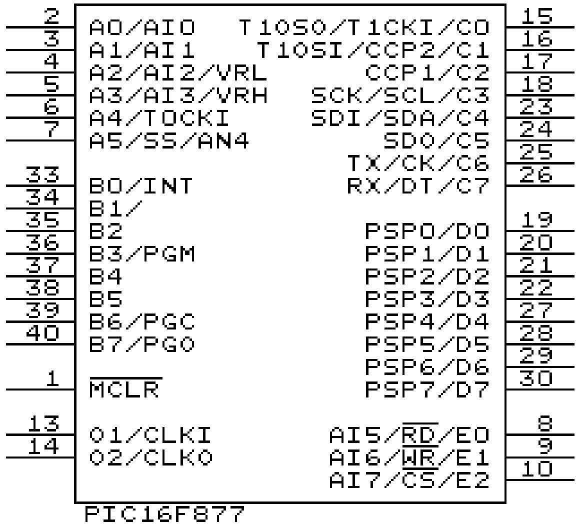
- PIC families
  - Packages: 8, 18, 20, 28, 40, 64, 84 pins
  - Built-in peripherials:
    - EEPROM
    - ADC, DAC
    - Analogue comparator
    - USART, SPI, I^{2}C, CCP
    - LIN, CAN, USB
    - Temperature sensor
    - Op-amp
    - …

PIC (1)

- PIC – example pinouts

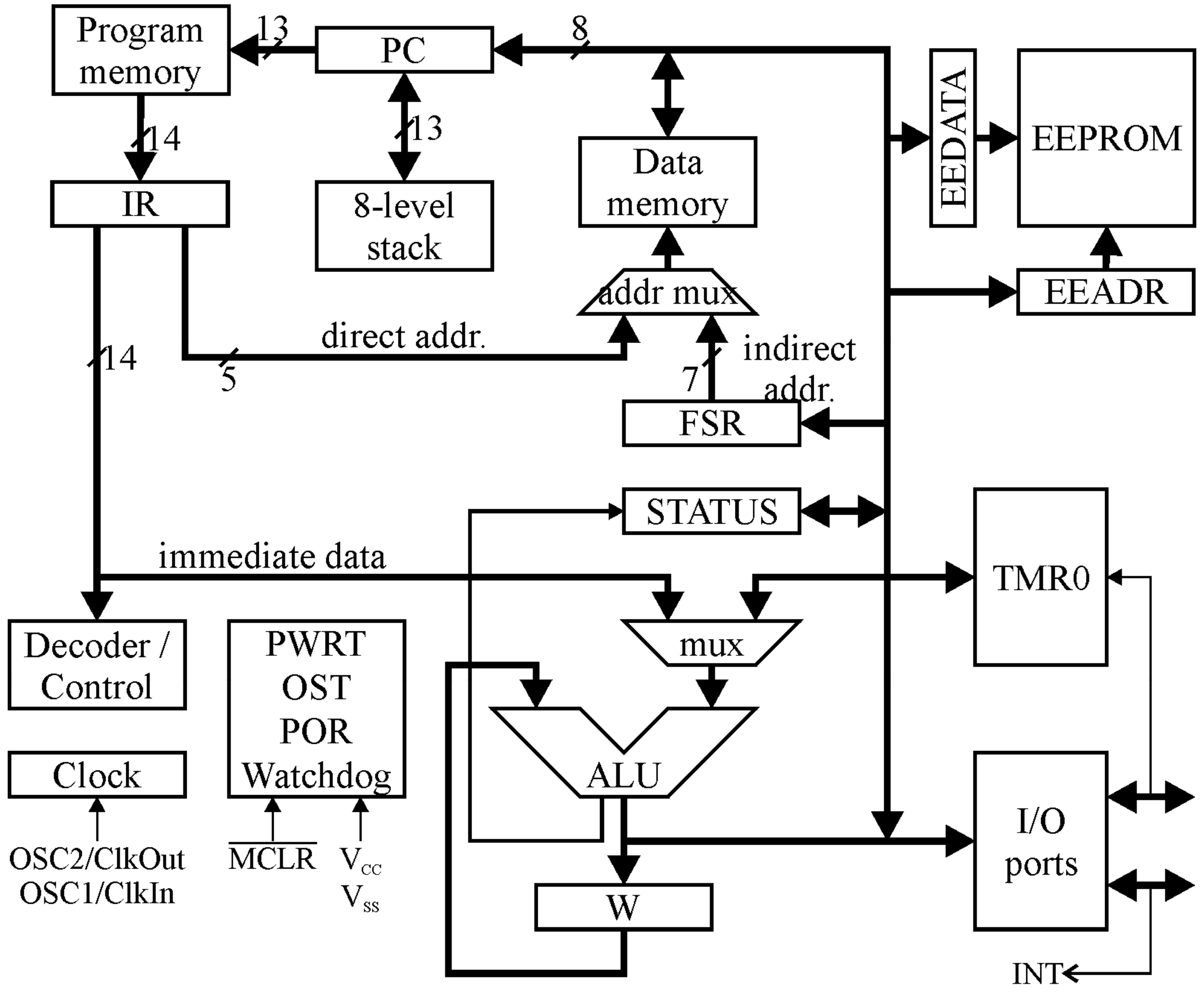






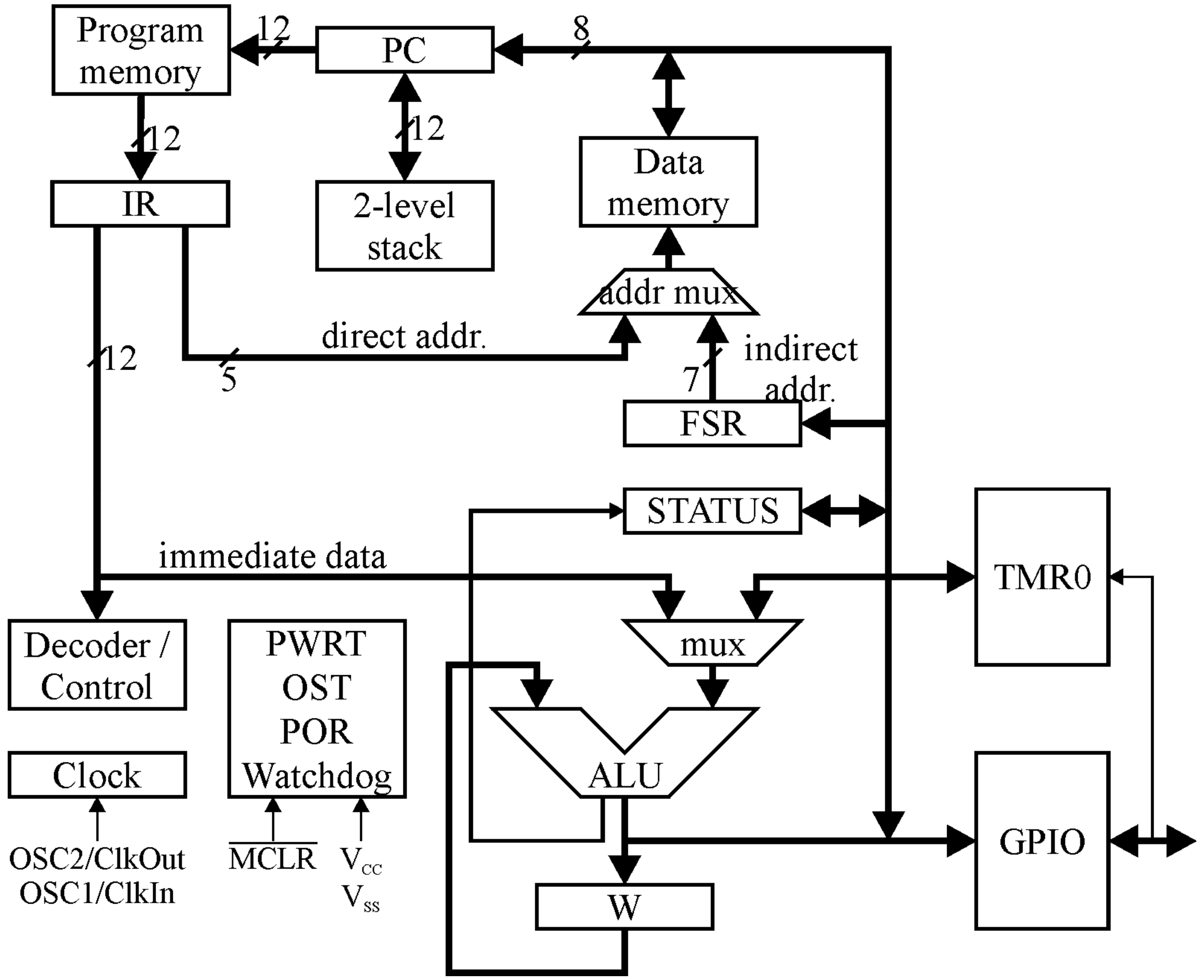
PIC (1)

- PIC16F8x – general architecture



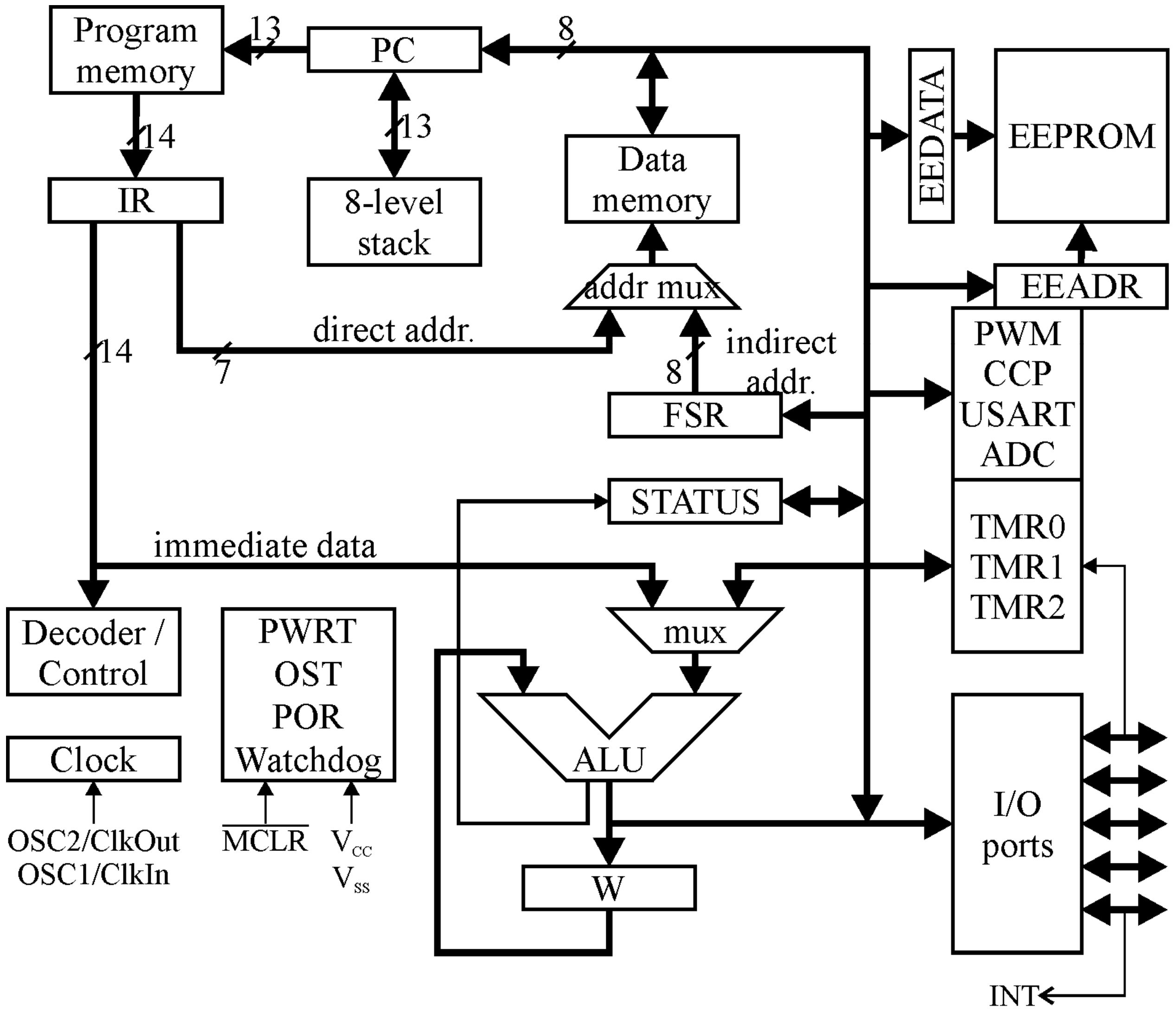
PIC (1)

- PIC12C509 – general architecture



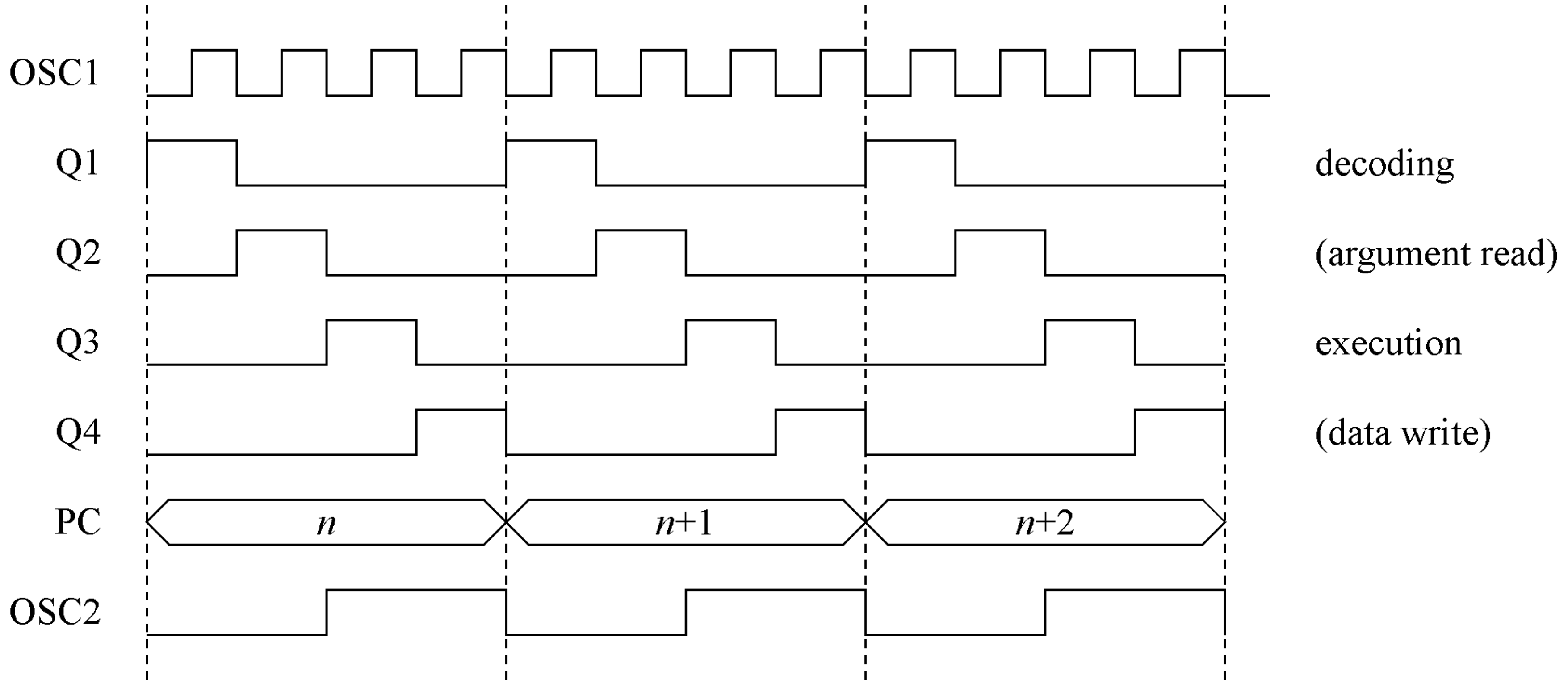
PIC (1)

- PIC16C877 – general architecture



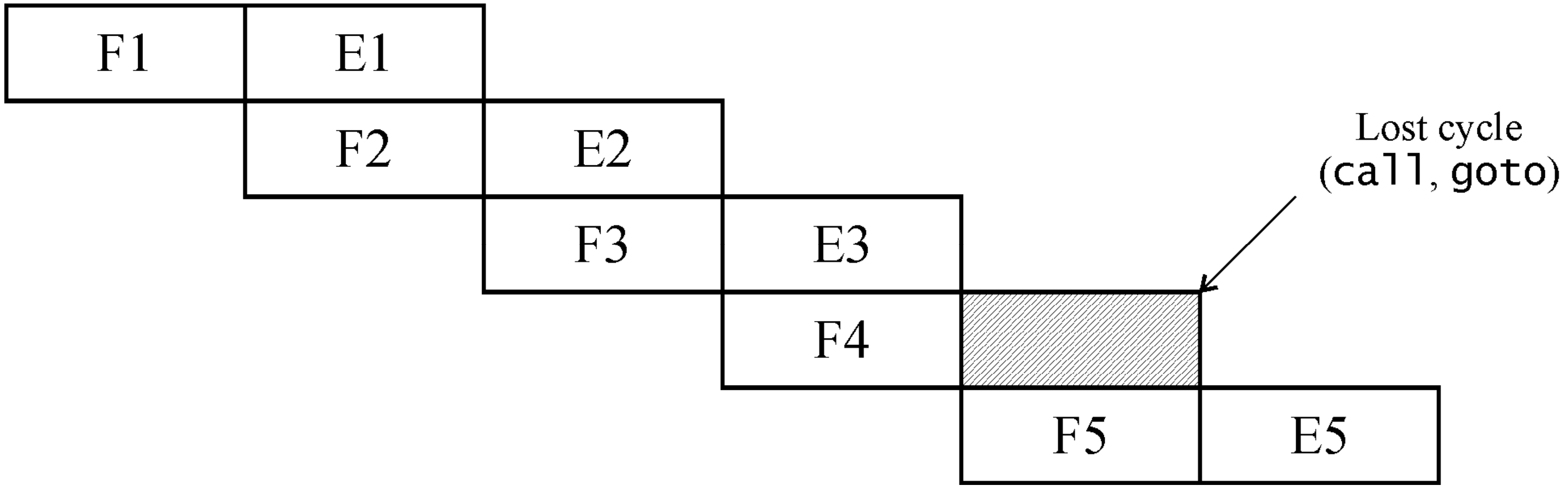
PIC (1)

- PIC16F8x – command cycle
  - 4 machine cycles Q1÷Q4
  - 1 machine cycle = 1 clock cycle
    - e.g., 4 MHz → 1µs/command



PIC (1)

- PIC16F8x – command cycle



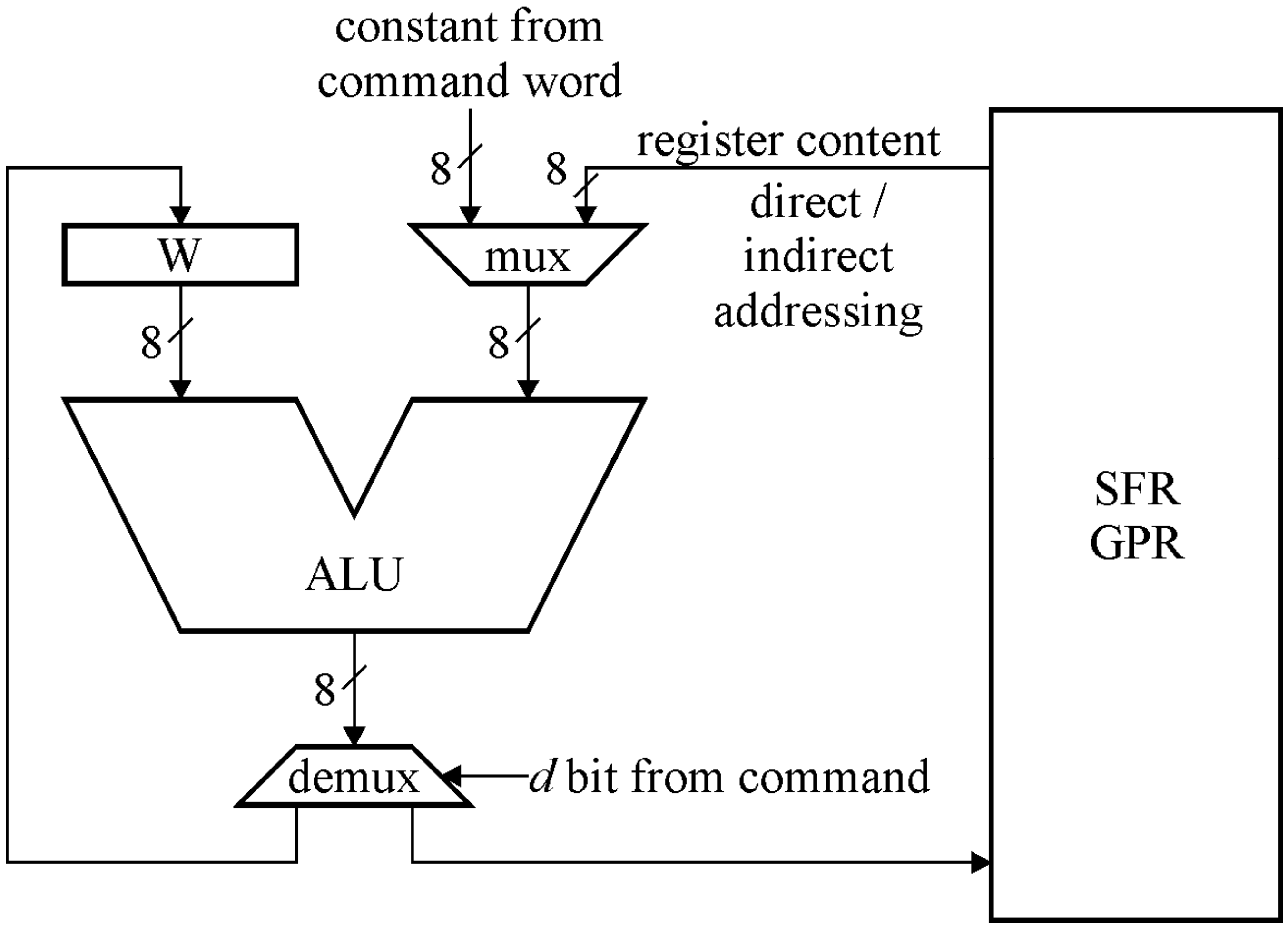
  - 1 command = 2 command cycles
    - Except `call`, `goto` & others operating on PC

PIC (1)

- PIC16F8x – ALU operations
  - Registers
    - W register (*accumulator or not?*)
    - GPR/SFR
  - ALU arguments/results
    - Arg1 $\leftarrow$ W
    - Arg2 $\leftarrow$ SFR, GPR or constant
    - Result $\rightarrow$ W, SFR, GPR
  - Flags modification (STATUS reg.)
    - C – carry/borrow
    - DC – auxilliary carry/borrow
    - Z – zero

PIC (1)

- PIC16F8x – ALU operations



PIC (1)

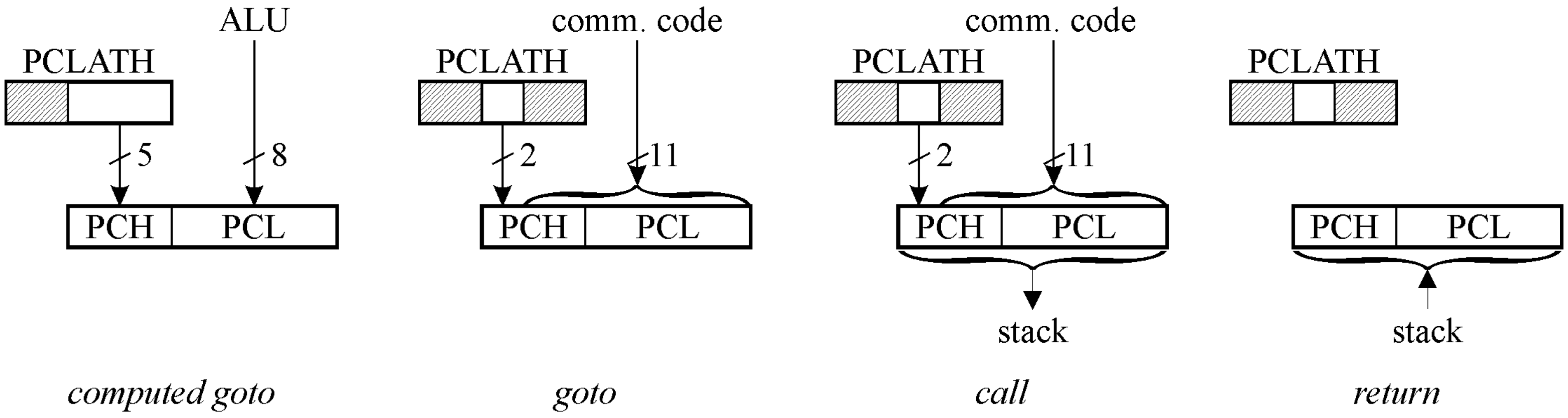
- PIC16F8x – ALU operations
  - STATUS register
    - Any value can be written
    - C, DC, Z auto set
    - !TO, !PD – non-modificable
      - !TO=0 → watchdog overflow
      - !PD=0 → after `sleep` command
    - IRP – RAM bank selection for indirect addressing
      - Unused if 2 banks only
    - RP0, RP1 – RAM bank selection for direct addressing
      - RP1 unused if less than 3 banks

PIC (1)

- PIC16F8x – OPTION_REG
  - !RBPU
    - 0 → PortB pull-up resistors on
  - INTEDG
    - Rising (1)/falling (0) edge external interrupt signalling
  - T0CS
    - Clock source for T0: external (1)/internal (0)
  - T0SE
    - Rising (1)/falling (0) edge active
  - PSA
    - Prescaler assigned to watchdog (1)/TMR0 (0)
  - PS2..0
    - Prescaler 1/2 … 1/256 (TMR0), 1/1 … 1/128 (watchdog)

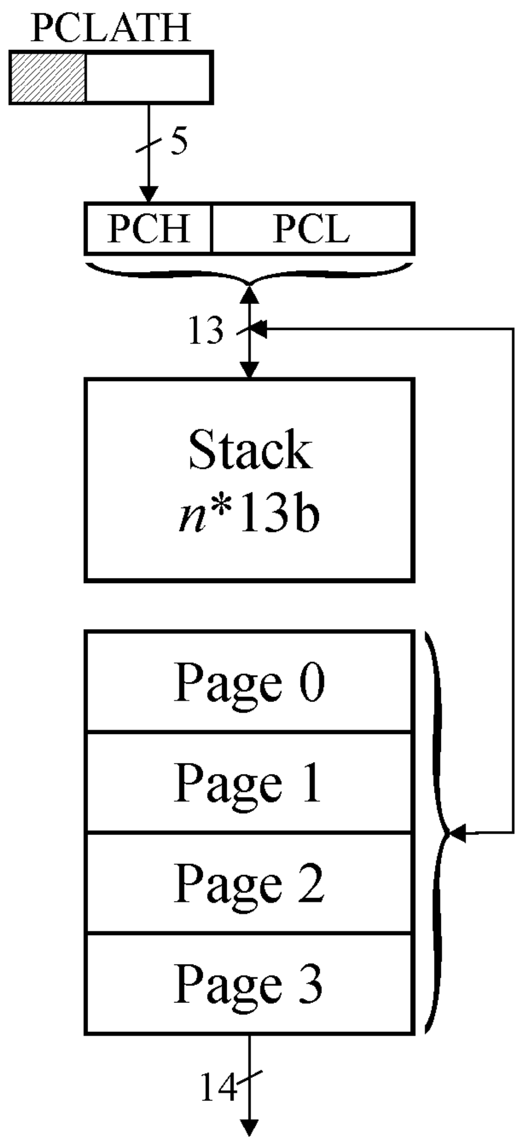
PIC (1)

- PIC16F8x – program counter (PC)
  - $PC_{0..12}$:
    - $PCL = PC_{0..7}$
      - Can be read or written
    - $PCH = PC_{8..12}$
      - Can't be directly modified
      - PCLATH for modification



PIC (1)

- PIC16F8x – program memory
  - Reset vector: 0000h
  - Interrupt vector: 0004h
  - Stack: hardware only
    - No software access
    - No `push`, `pop` commands
    - No argument passing
    - **No overflow signalling!**
  - Max capacity = 8k×14
    - Can be less than 8k

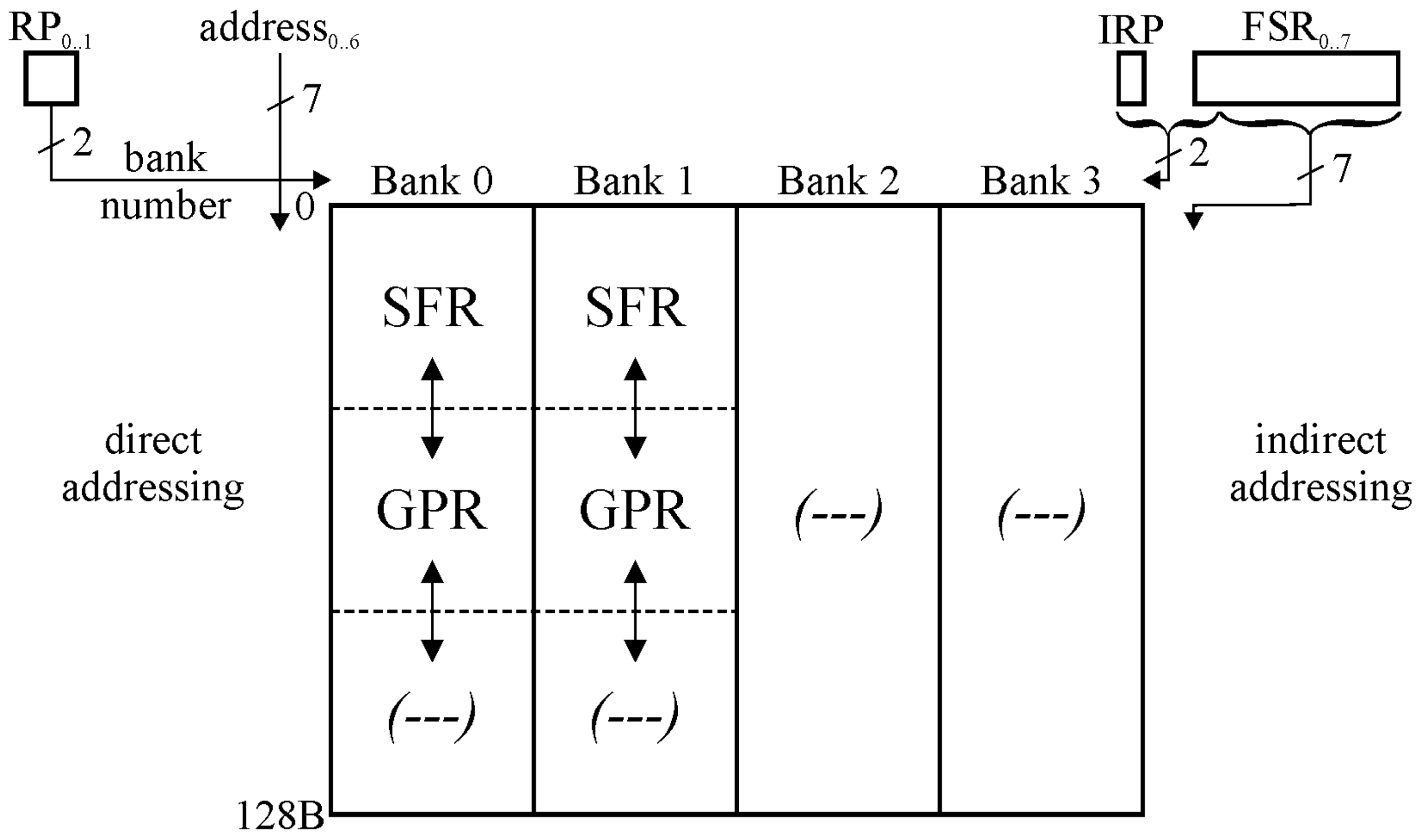


PIC (1)

- PIC16F8x – configuration memory
  - Above program memory
  - Available only during programming
  - Options:
    - Oscillator type
    - Watchdog on/off
    - Power-on delay
    - Program/EEPROM read disable
      - Can’t be read from outside of the µp

PIC (1)

- PIC16F8x – data memory

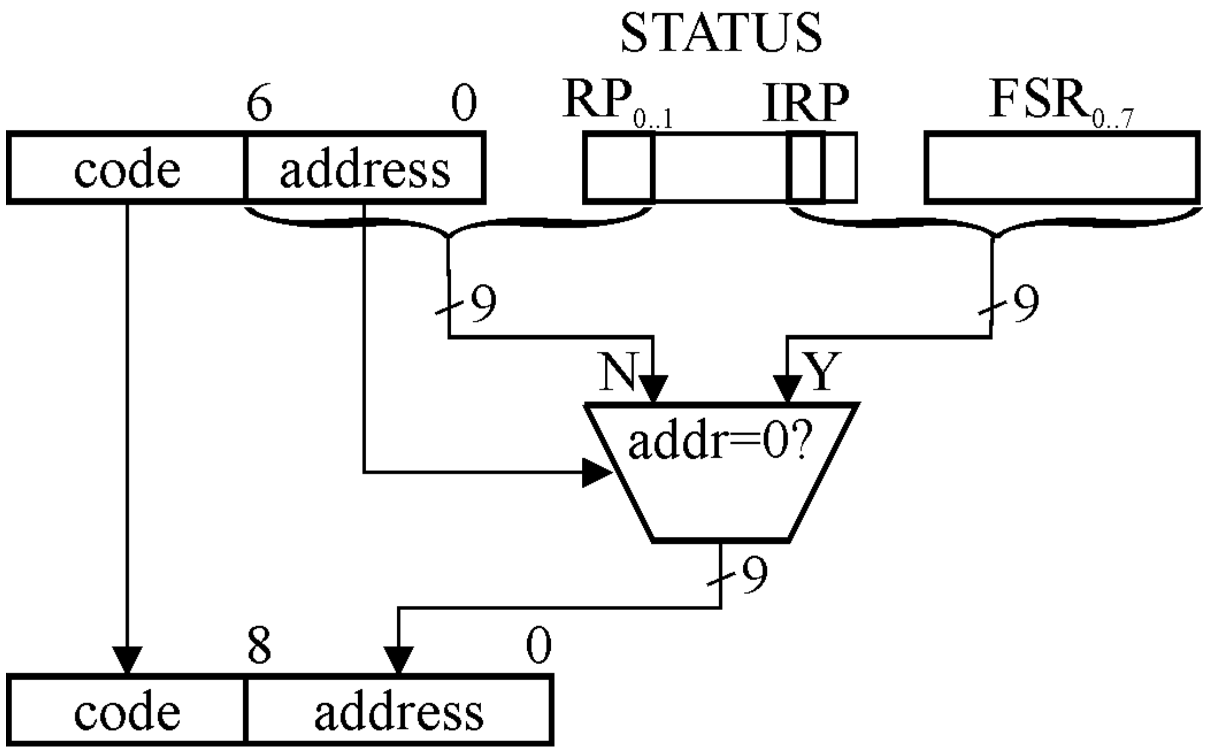


PIC (1)

- PIC16F8x – data memory
  - μc-dependent:
    - Capacity, zone borders, banks number
  - SFR:
    - Key registers
      - Mapped in each bank
      - Constant address
    - Remaining registers
      - Mapped in a single bank only
  - GPR (bank 1) and GPR (bank 0)
    - May map to the same memory cells

PIC (1)

- PIC16F8x – data memory addressing



  – Direct INDF access = indirect GPR access

PIC (1)

- PIC16F8x – data memory addressing
  - Indirect addressing
    - INDF pseudo-register
      - Mapped at address 0 in each bank
      - Is not a real register
    - Read:
      - addr → IRP.FSR
      - INDF → data
    - Write
      - Addr → IRP.FSR
      - Data → INDF

PIC (1)

- PIC16F8x – EEPROM memory
  - Only indirect-addressable
  - $\geq 10^7$ erase-write cycles
  - Registers:
    - EEDATA – 8-bit data, Rd/Wr
    - EEADR – 8-bit address
    - EECON1 – Rd/Wr control
      - WR, RD, WREN, WRERR, EEIF, EEPGD
    - EECON2 – write „magic word”

PIC (1)

- PIC16F8x – EEPROM memory

Read	Write
addr → EEADR	addr → EEADR
EEPGD = 0	data → EEDATA
EECON1.RD = 1	(EEPGD = 0)
(Δt = 0)	EECON1.WREN = 1
EEDATA → data	EECON2 = 55h
	EECON2 = AAh
	EECON1.WR = 1
	(Δt ≈ 10 ms)
	EEIF = 1, EECON1.WR cleared

PIC (1)

- PIC16F8x – FLASH memory
  - Similar to EEPROM, but longer address & data reg.
  - Write only to non-secured area

Read	Write
addr → EEADR, EEADRH	addr → EEADR, EEADRH
EEPGD = 1	data → EEDATA, EEDATH
EECON1.RD = 1	(EEPGD = 1)
nop *Δt for read from*	EECON1.WREN = 1
nop *program memory*	EECON2 = 55h
EEDATA, EEDATH → data	EECON2 = AAh
	EECON1.WR = 1
	nop *μp omits this*
	nop *Δt ≈ 10 ms*
	EEIF = 1, EECON1.WR cleared

  - Programming from remote μc by „boot-loader”