MEMORY & I/O SYSTEMS

Chapter 8

Digital Design and Computer Architecture, 2nd Edition

David Money Harris and Sarah L. Harris

ELSEVIER

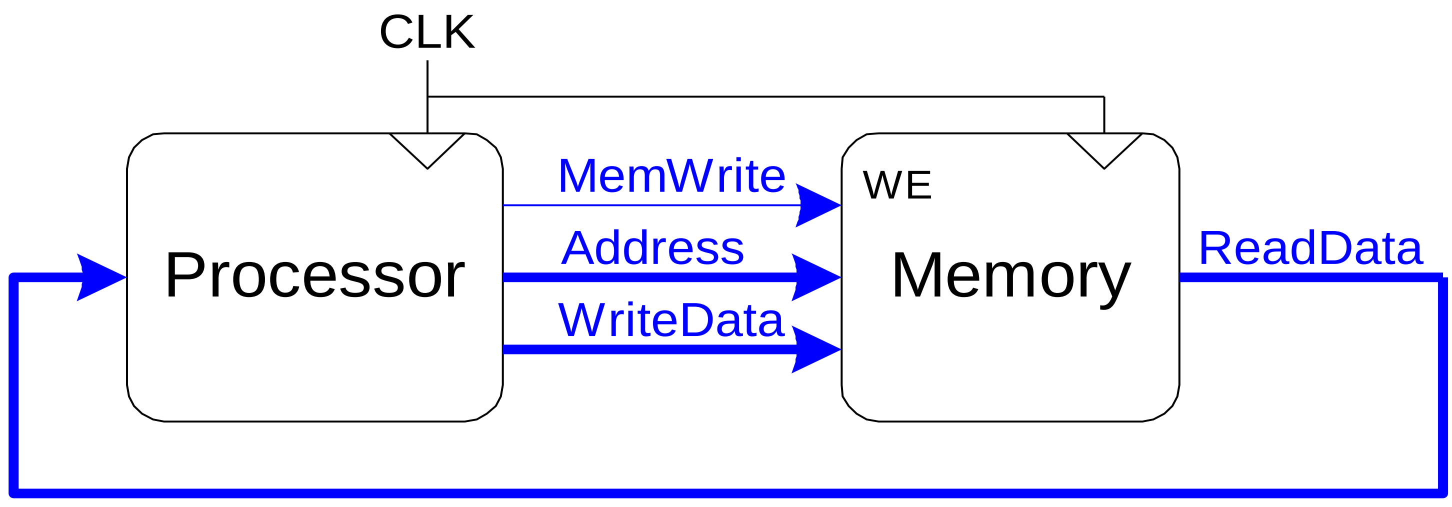
Memory-Mapped I/O

- Processor accesses I/O devices just like memory (like keyboards, monitors, printers)
- Each I/O device assigned one or more address
- When that address is detected, data read/written to I/O device instead of memory
- A portion of the address space dedicated to I/O devices

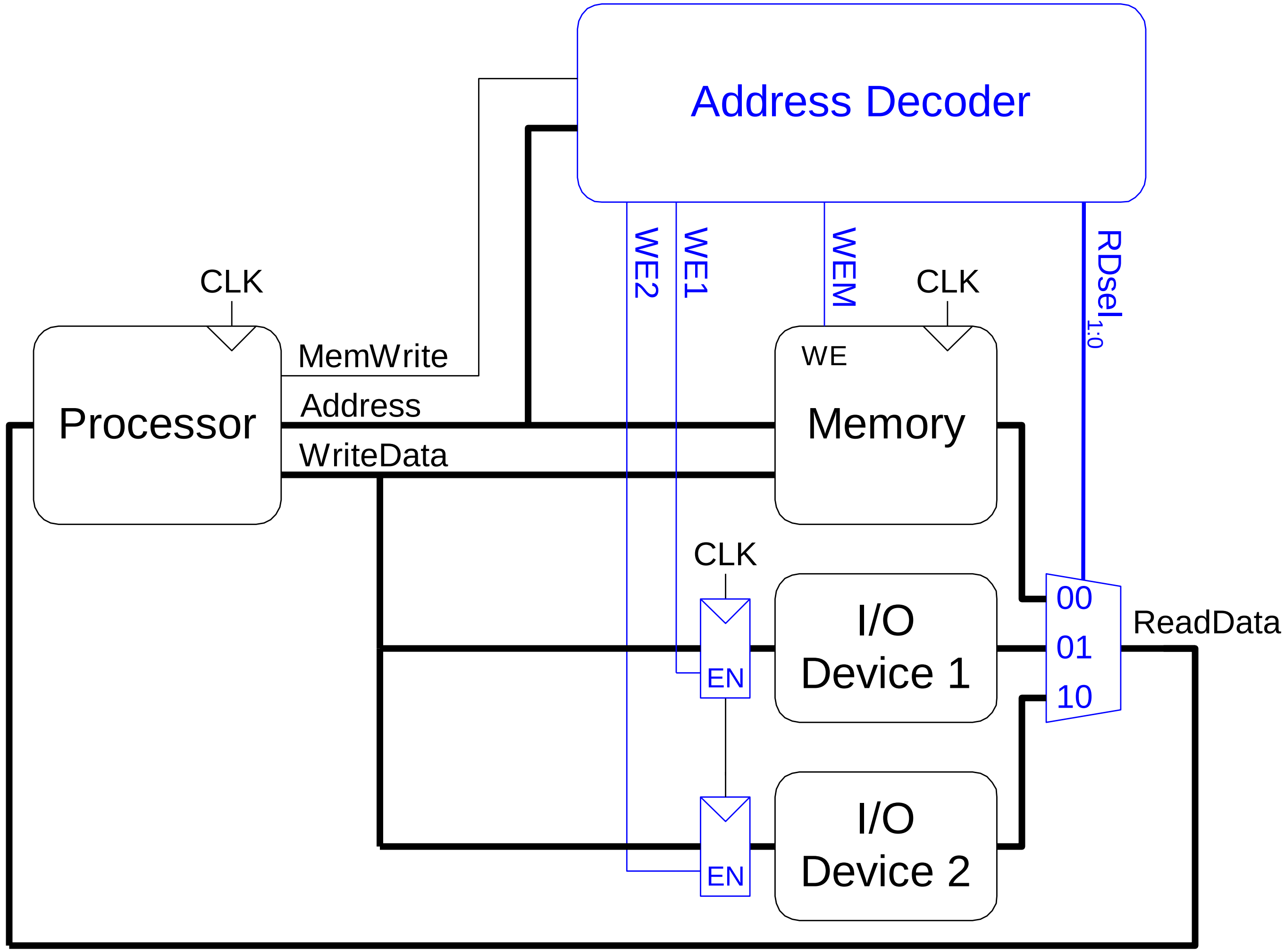
Memory-Mapped I/O Hardware

- **Address Decoder:**
  - Looks at address to determine which device/memory communicates with the processor
- **I/O Registers:**
  - Hold values written to the I/O devices
- **ReadData Multiplexer:**
  - Selects between memory and I/O devices as source of data sent to the processor

The Memory Interface



Memory-Mapped I/O Hardware



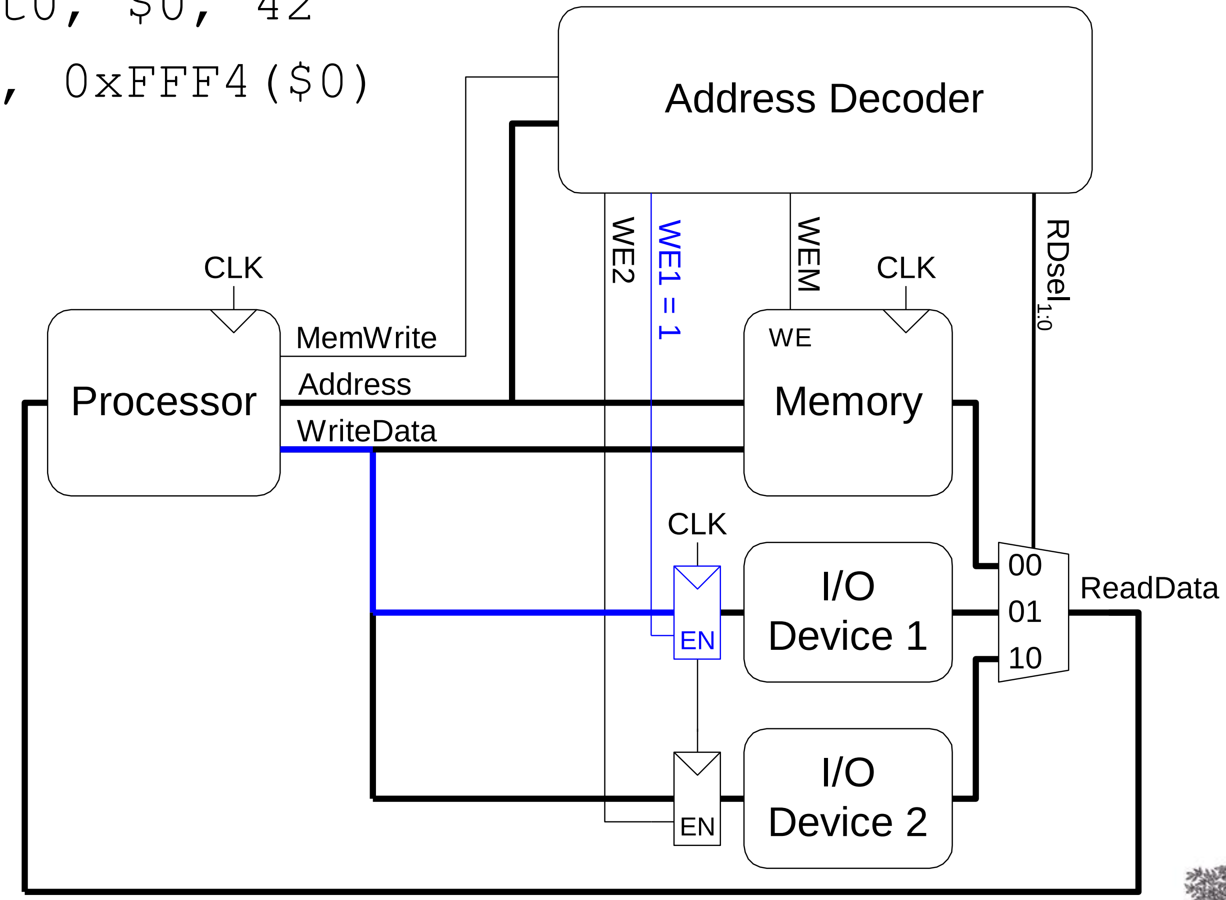
Memory-Mapped I/O Code

- Suppose I/O Device 1 is assigned the address 0xFFFFFFF4
  - **Write the value 42 to I/O Device 1**
  - **Read value from I/O Device 1 and place in `$t3`**

Memory-Mapped I/O Code

- **Write the value 42 to I/O Device 1 (0xFFFFFFF4)**

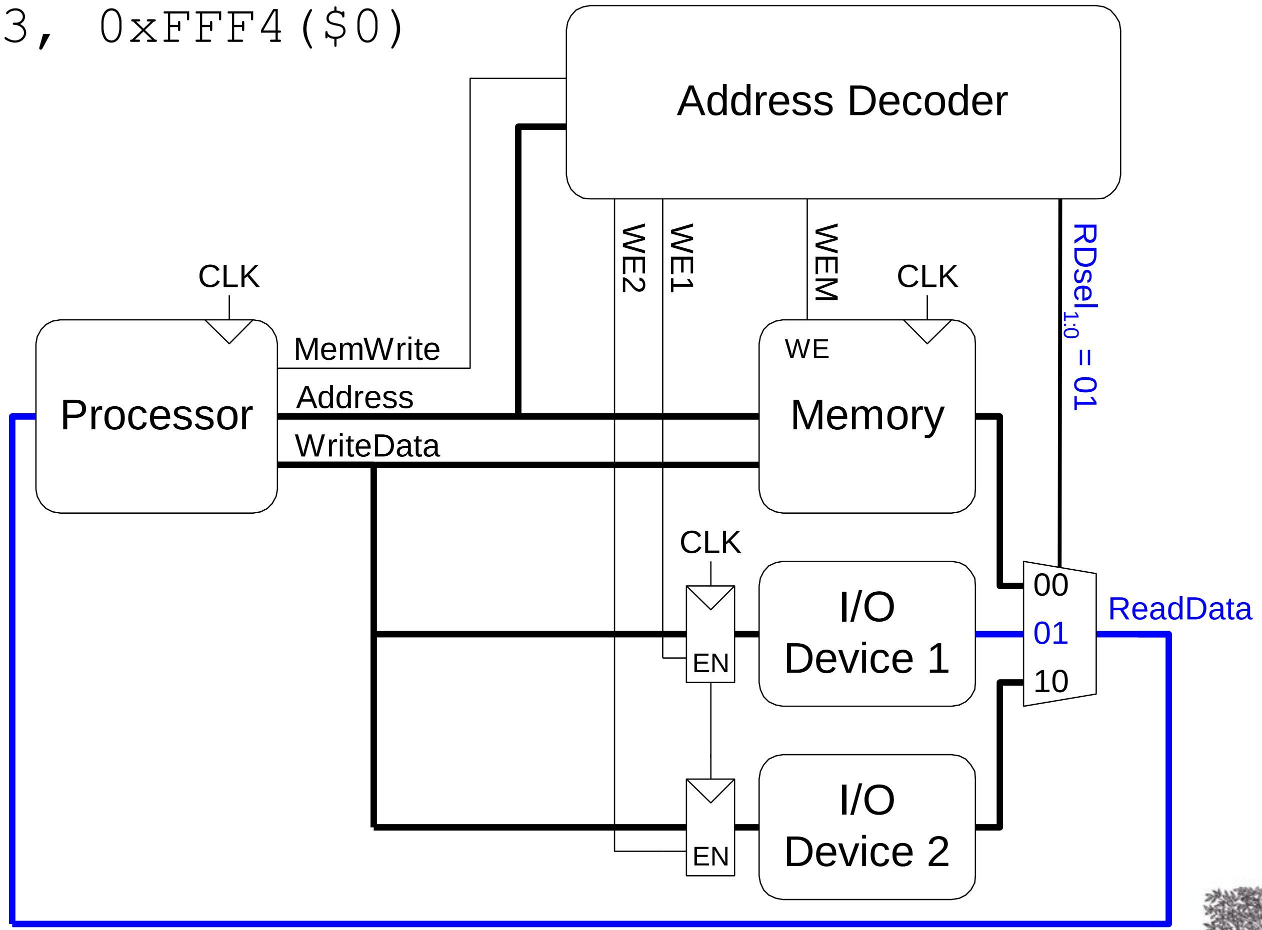
```
addi $t0, $0, 42
sw $t0, 0xFFF4($0)
```



Memory-Mapped I/O Code

- **Read the value from I/O Device 1 and place in `$t3`**

```
lw $t3, 0xFFF4($0)
```



Input/Output (I/O) Systems

- Embedded I/O Systems
  - Toasters, LEDs, etc.
- PC I/O Systems

Embedded I/O Systems

- Example microcontroller: PIC32
  - microcontroller
  - 32-bit MIPS processor
  - low-level peripherals include:
    - serial ports
    - timers
    - A/D converters

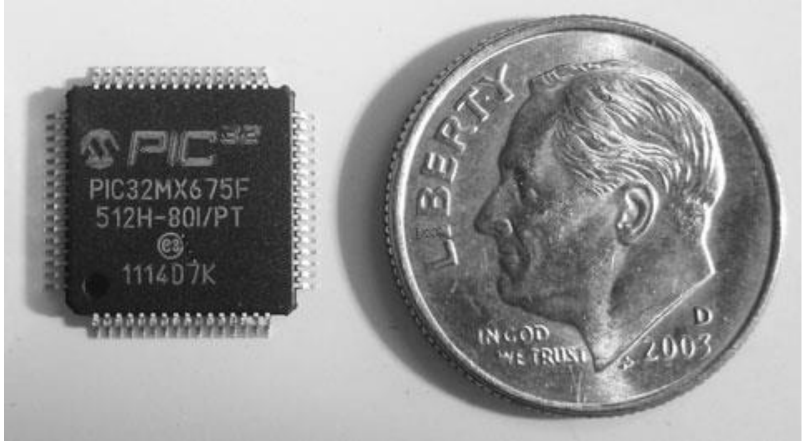
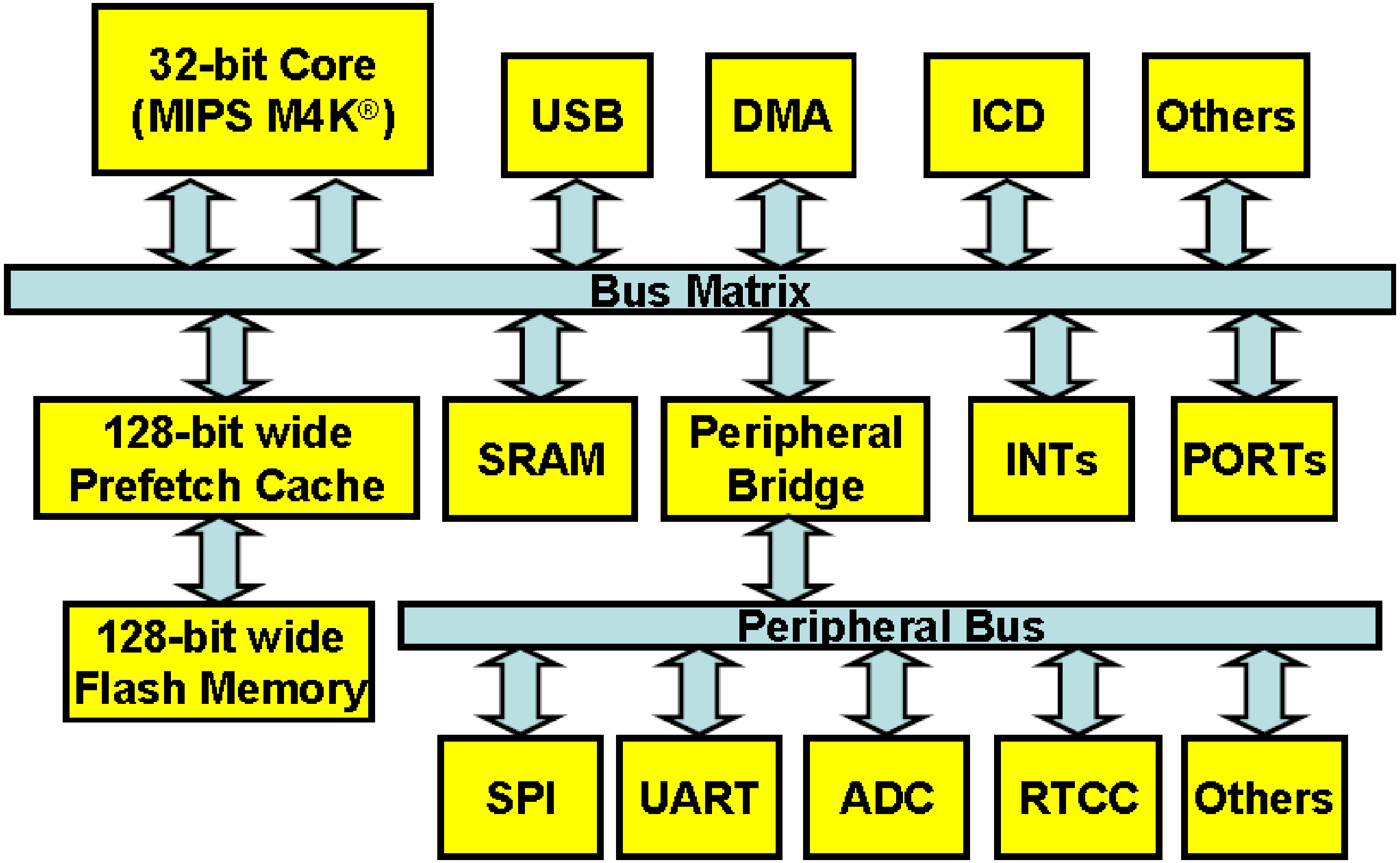
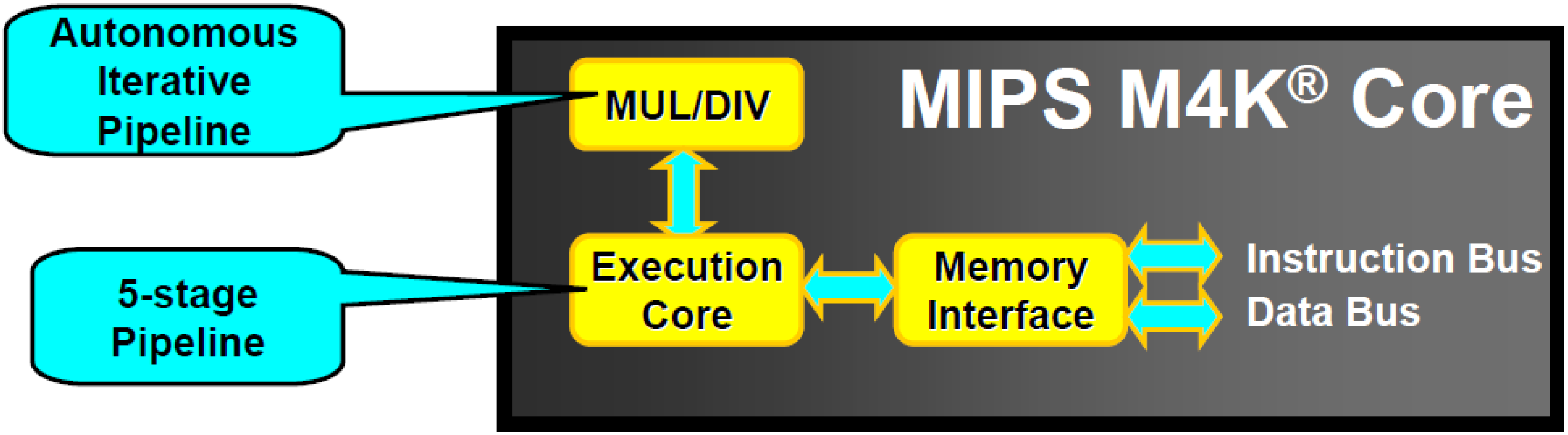


Figure 8.32 PIC32 in 64-pin TQFP package

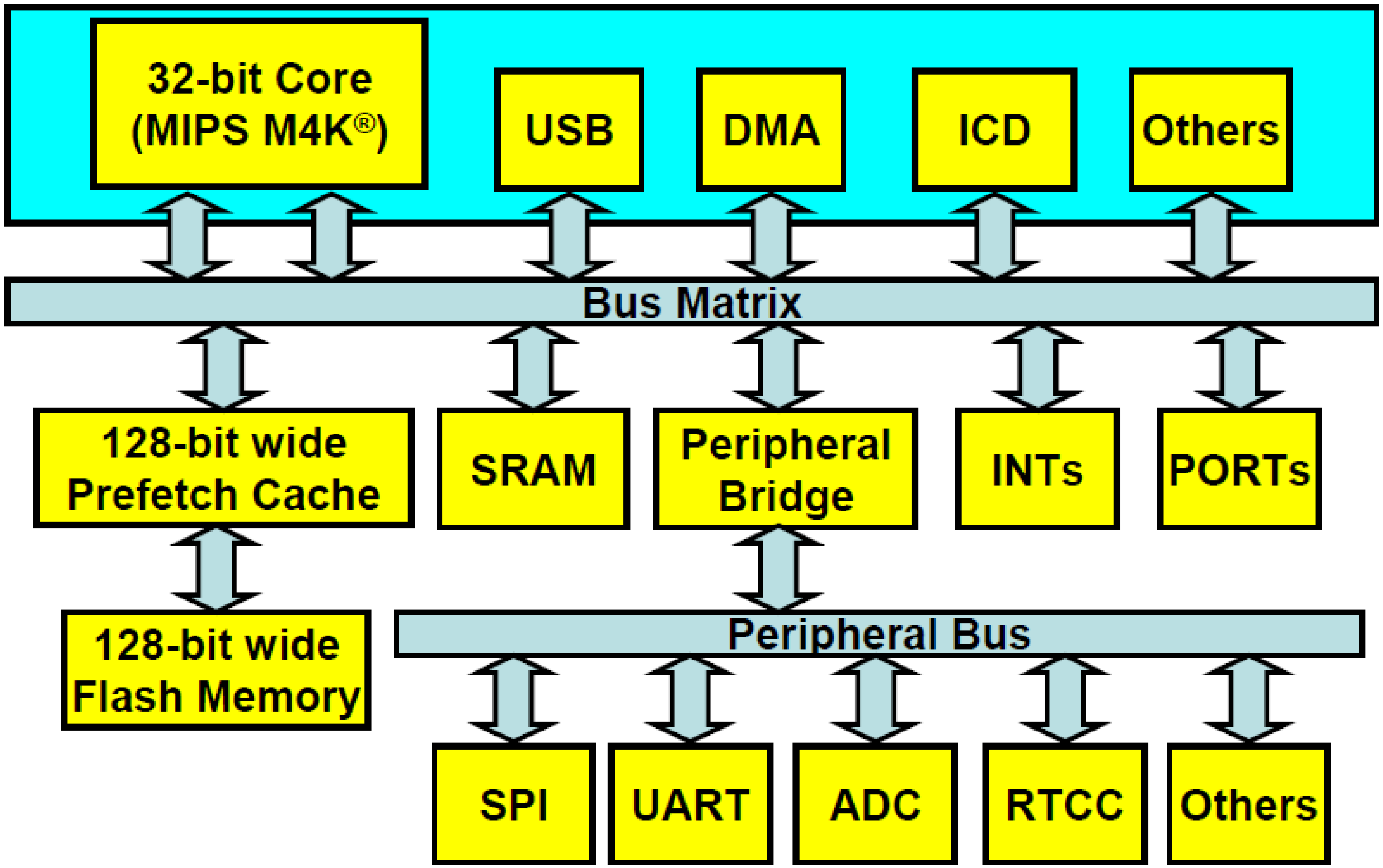
PIC32


32-bit Core
(MIPS M4K®)
USB
DMA
ICD
Others
Bus Matrix
128-bit wide
Prefetch Cache
SRAM
Peripheral
Bridge
INTs
PORTs
128-bit wide
Flash Memory
Peripheral Bus
SPI
UART
ADC
RTCC
Others

PIC32 Core

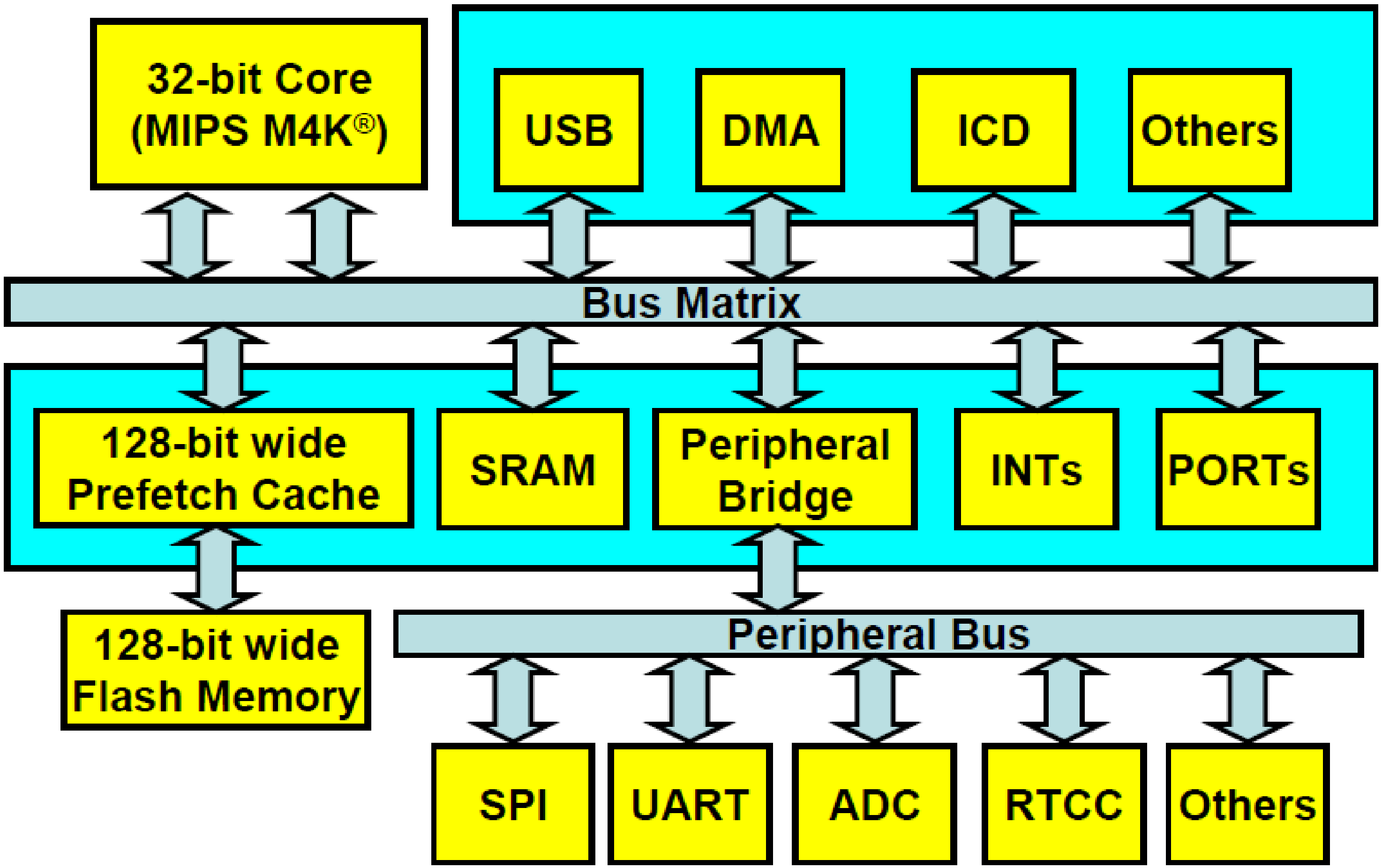

Autonomous Iterative Pipeline
5-stage Pipeline
MIPS M4K® Core
MUL/DIV
Execution Core
Memory Interface
Instruction Bus
Data Bus

Bus Masters

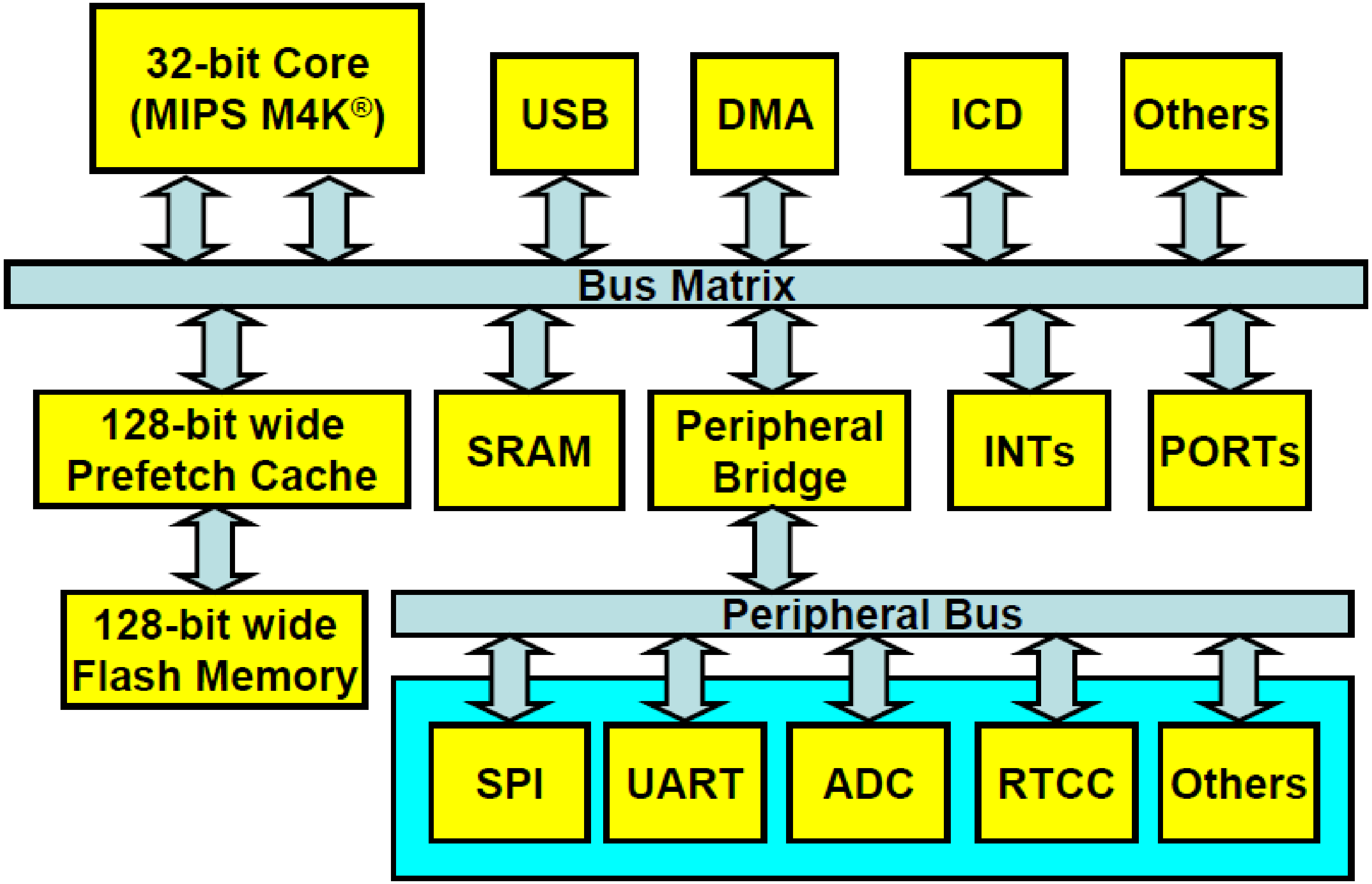


ELSEVIER

Sysclk Peripherals



Pbclk Peripherals



PIC32 Block Diagram

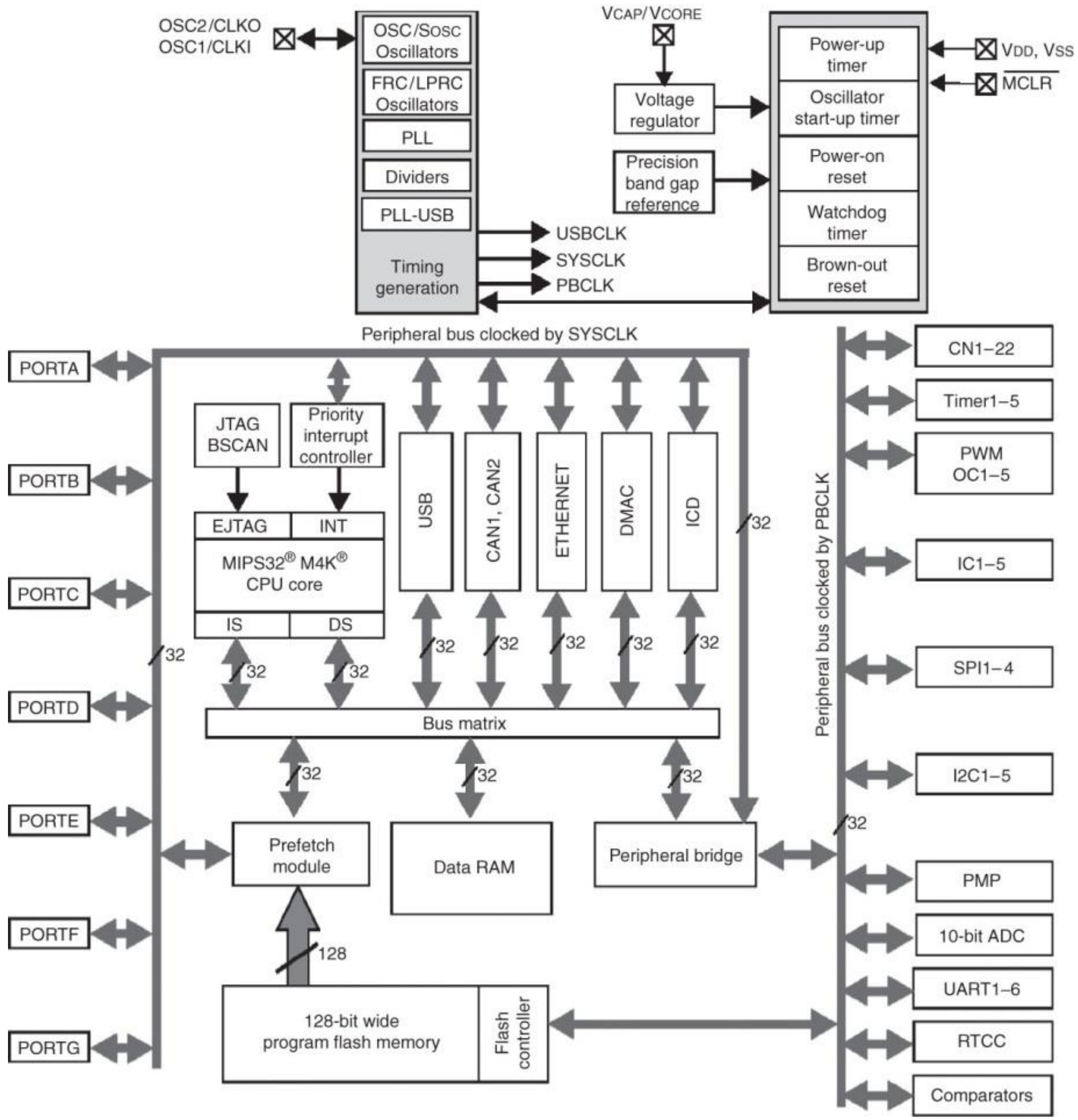


Figure 8.29 PIC32 block diagram

PIC32 Internals

- The core connects to the rest of the modules via Bus Matrix. The Bus Matrix is a high-speed switch.
  - Point to point connection between modules. CPU core, USB and
  - DMA connect to the SRAM, SPI, UART, etc., via the
  - The Bus Matrix runs at the same speed as the CPU, while the Peripheral Bus can be programmed to run at a different clock than the CPU. The exact Bus clock is determined by the Peripheral Bridge setting.
- PIC32 uses a 128-bit wide Flash memory
  - Instruction throughput and improve CPU performance
  - A 128-bit Prefetch Cache module - next 128-bits of instructions
  - CPU is running faster than Flash memory speed.

PIC32 Memory Map

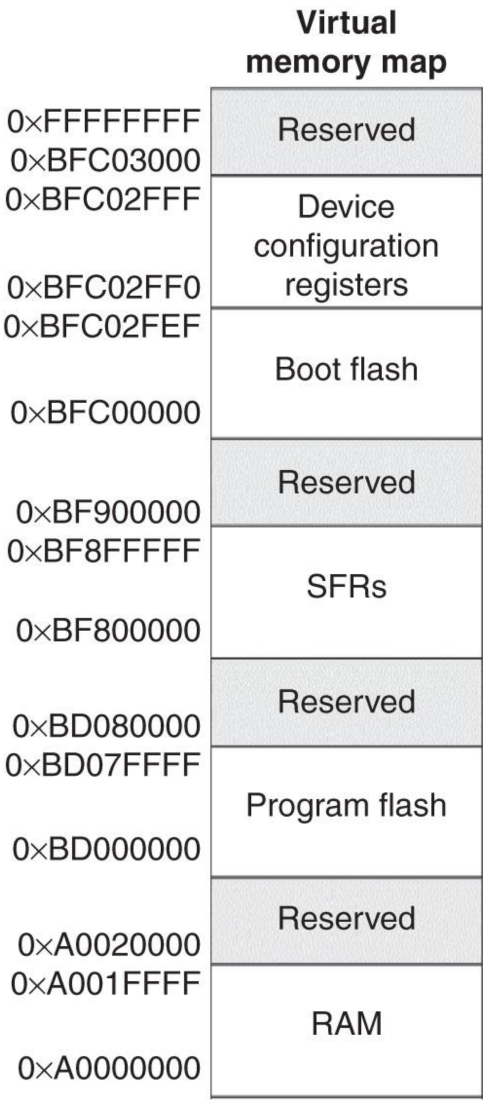


Figure 8.30 PIC32 Memory Map

PIC32 Pinout Diagram

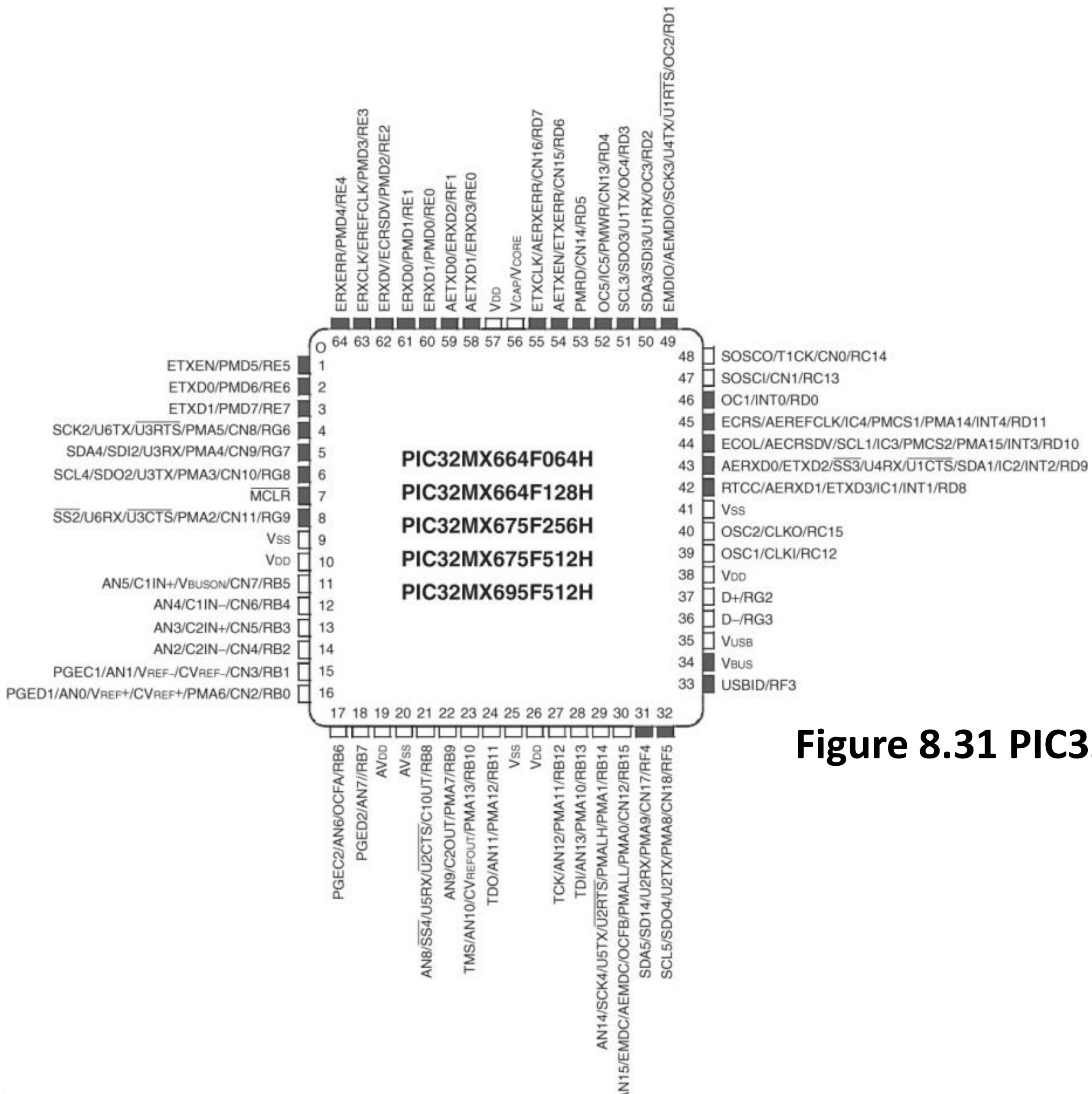


Figure 8.31 PIC32 pinout

PIC32 Operational Diagram

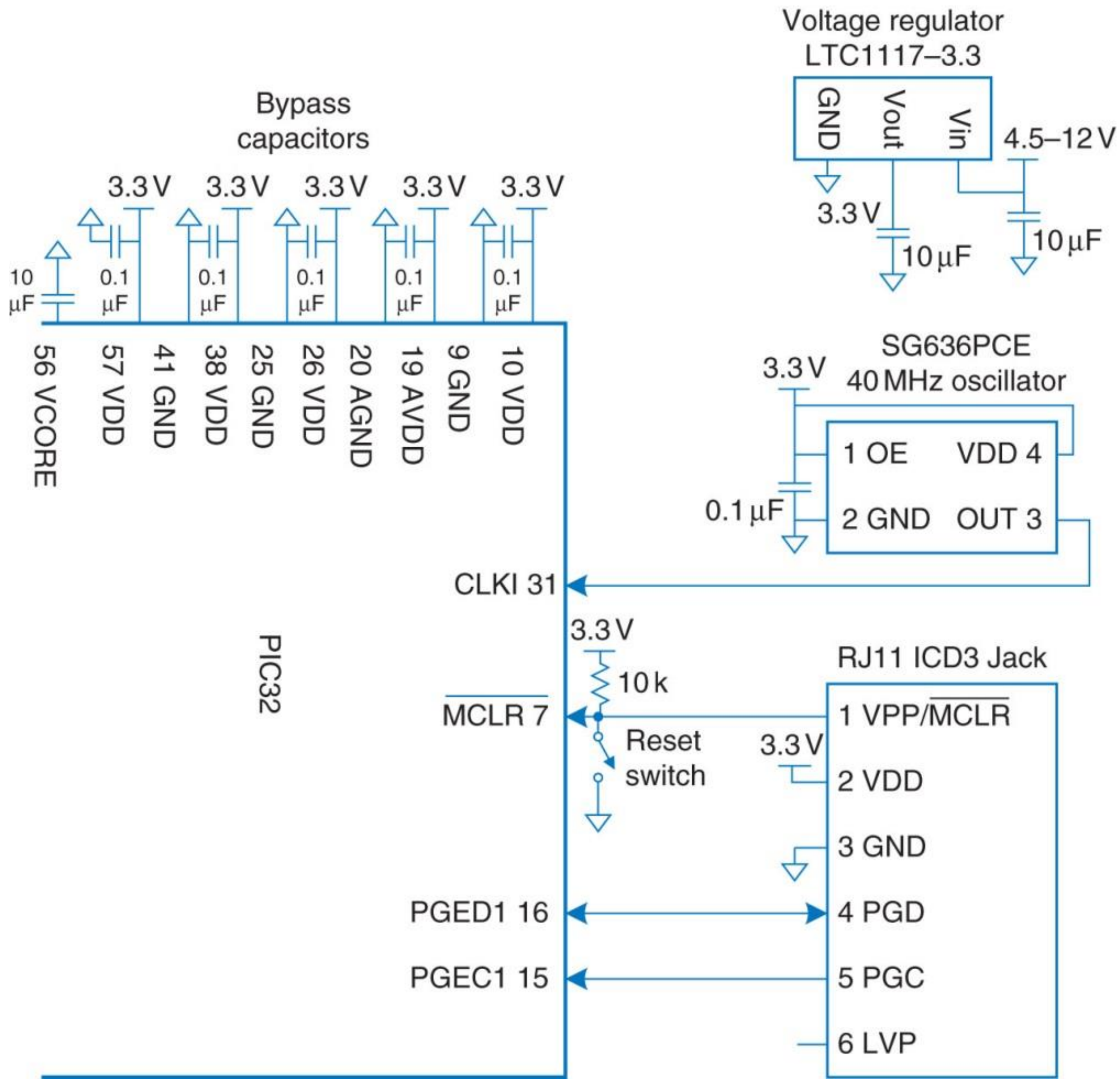


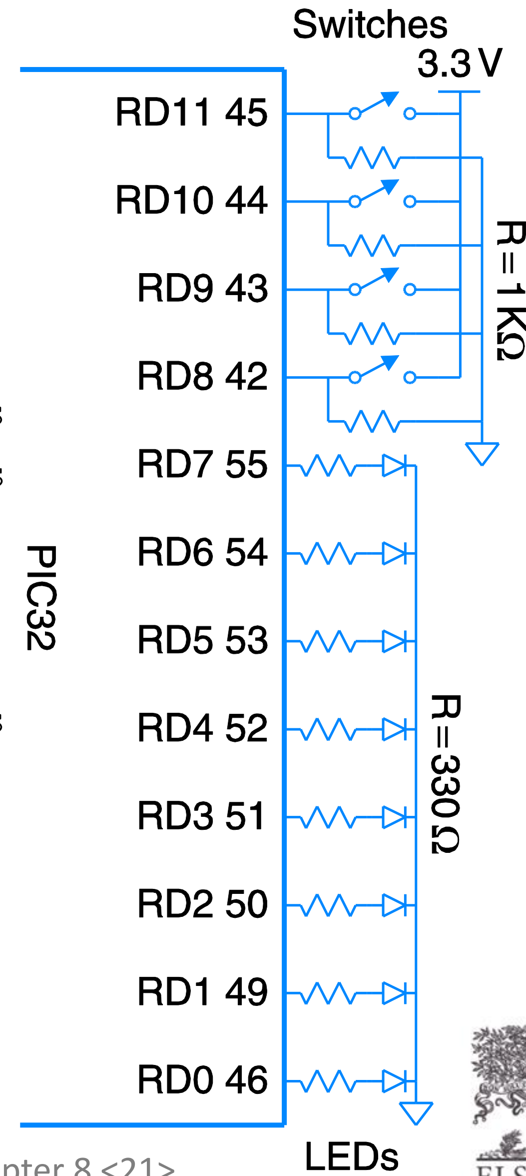
Figure 8.33 PIC32 basic operational schematic

MEMORY & I/O SYSTEMS

ELSEVIER

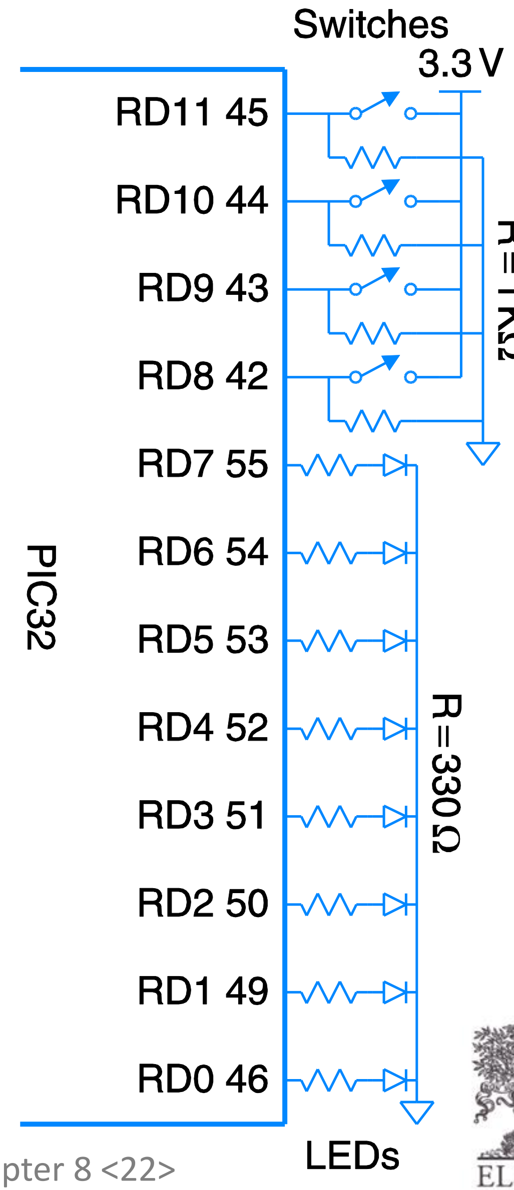
Digital I/O

```c
// C Code
#include <p3xxxx.h>

int main(void) {
  int switches;
  TRISD = 0xFF00;         // RD[7:0] outputs
                          // RD[11:8] inputs
  while (1) {
    // read & mask switches, RD[11:8]
    switches = (PORTD >> 8) & 0xF;
    PORTD = switches;  // display on LEDs
  }
}
```



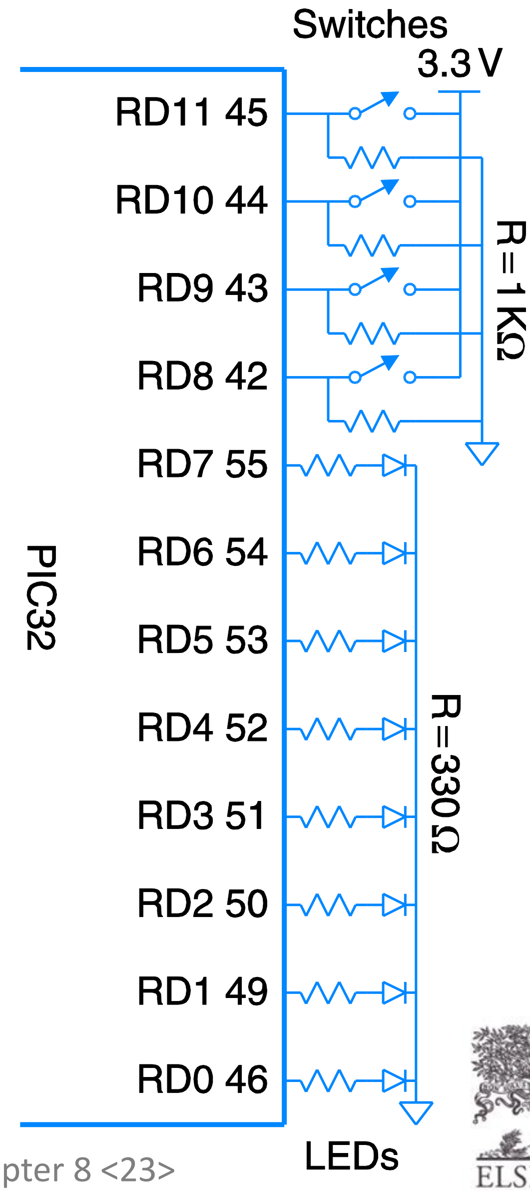
Setting GPIO Bits

```
// C Code
#include <p3xxxx.h>

int main(void) {
  int switches;

    while (1) {
      PORTDbits.RD0 = PORTDbits.RD8;
      // Copy the value of first
      // switch to first Led
    }
}
```



Set & Clr

```
// C Code
#include <p3xxxx.h>

int main(void) {
  int switches;

     while (1) {
        PORTDSET = 0b0101;
        // Set -> 1
        PORTDCLR = 0b1000;
        // Clr -> 0
        // 1110 becomes 0111

     }
}
```



GPIO

- The number of pins available depends on the size of the package of PIC
- Some or only input
  - RG[3:2]
- Some are multiple function
  - RB[15:0] shared as analog inputs

PORT vs. LAT

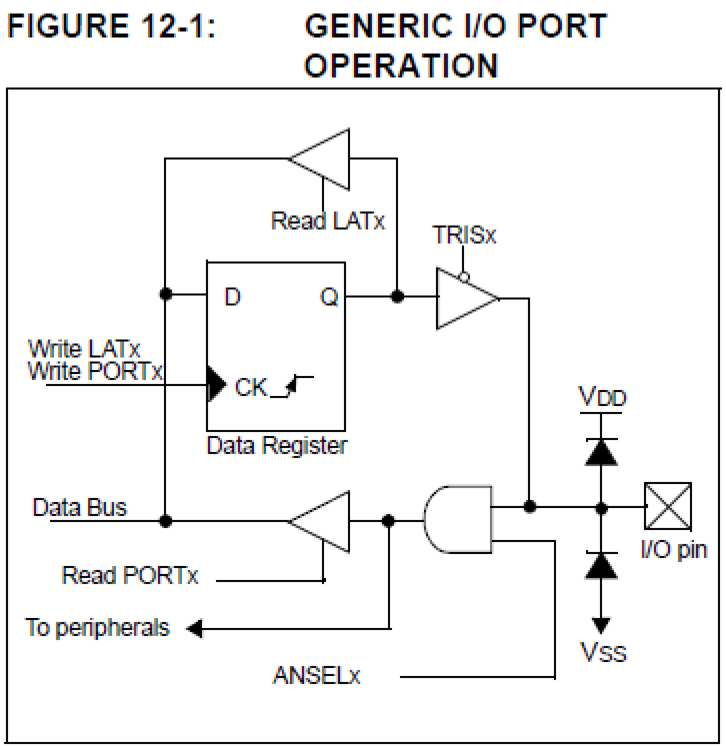


FIGURE 12-1: GENERIC I/O PORT OPERATION
Read LATx
TRISx
D
Q
Write LATx
Write PORTx
CK
Data Register
VDD
Data Bus
I/O pin
Read PORTx
To peripherals
VSS
ANSELx

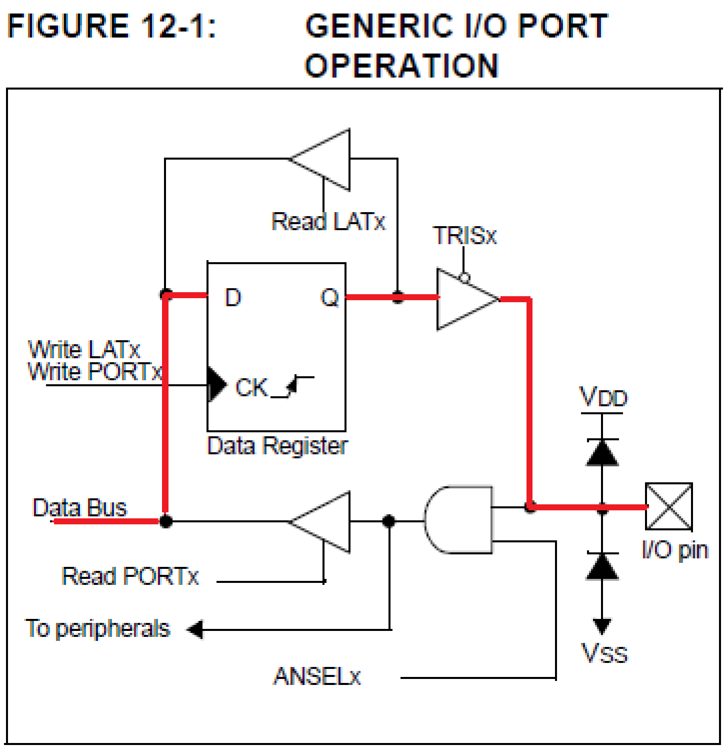


FIGURE 12-1: GENERIC I/O PORT OPERATION
Read LATx
TRISx
D
Q
Write LATx
Write PORTx
CK
Data Register
VDD
Data Bus
I/O pin
Read PORTx
To peripherals
VSS
ANSELx

PORT vs. LAT

FIGURE 12-1: GENERIC I/O PORT OPERATION

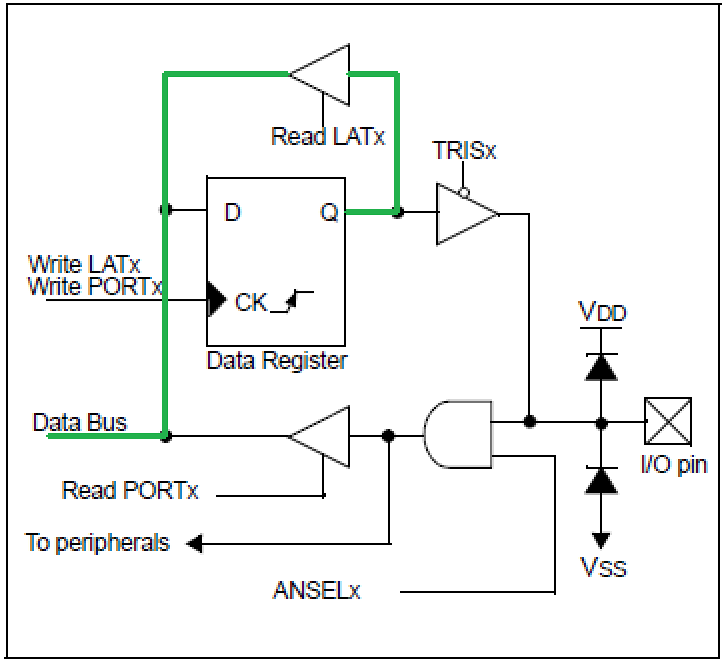
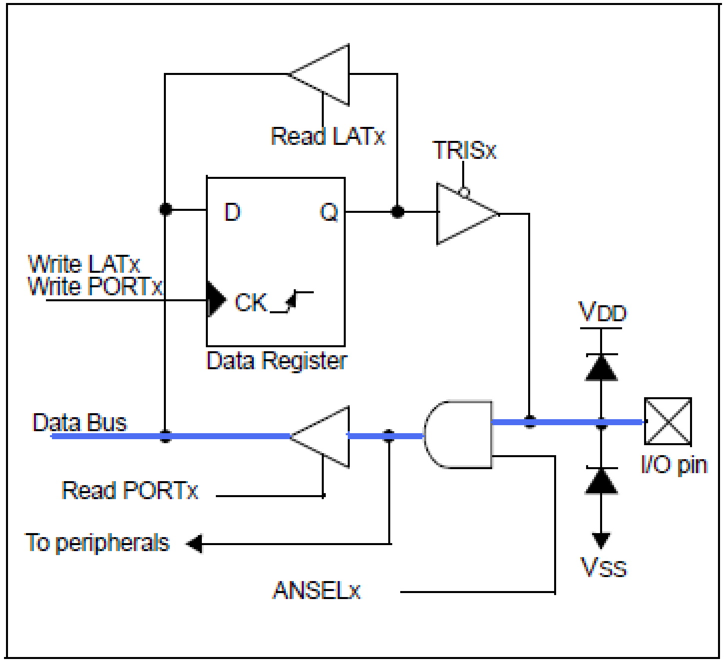


FIGURE 12-1: GENERIC I/O PORT OPERATION

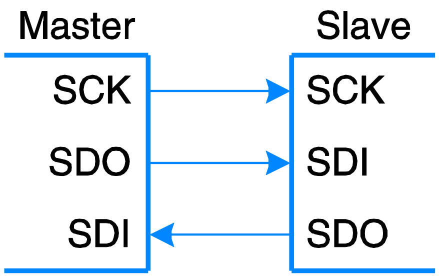


Serial I/O

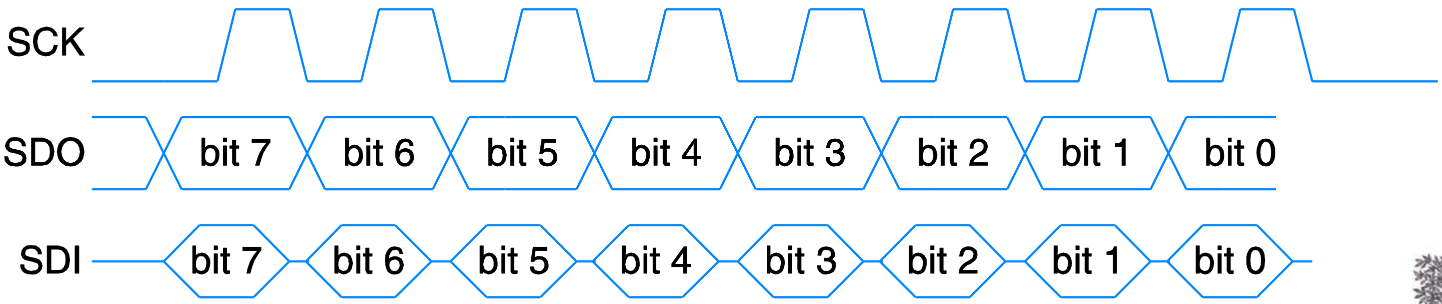
- Example serial protocols
  - **SPI:** Serial Peripheral Interface
  - **UART:** Universal Asynchronous Receiver/Transmitter
  - Also: I^2C, USB, Ethernet, etc.

SPI: Serial Peripheral Interface

- Master initiates communication to slave by sending pulses on SCK
- Master sends SDO (Serial Data Out) to slave, msb first
- Slave may send data (SDI) to master, msb first



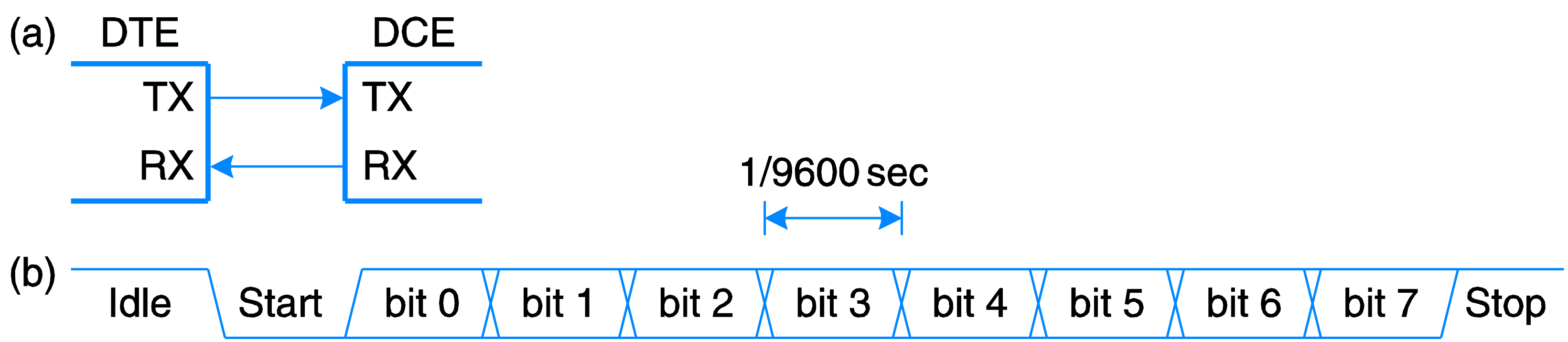
(a)



(b)

UART: Universal Asynchronous Rx/Tx

- Configuration:
  - start bit (0), 7-8 data bits, parity bit (optional), 1+ stop bits (1)
  - data rate: 300, 1200, 2400, 9600, …115200 baud
- Line idles HIGH (1)
- Common configuration:
  - 8 data bits, no parity, 1 stop bit, 9600 baud



Timers

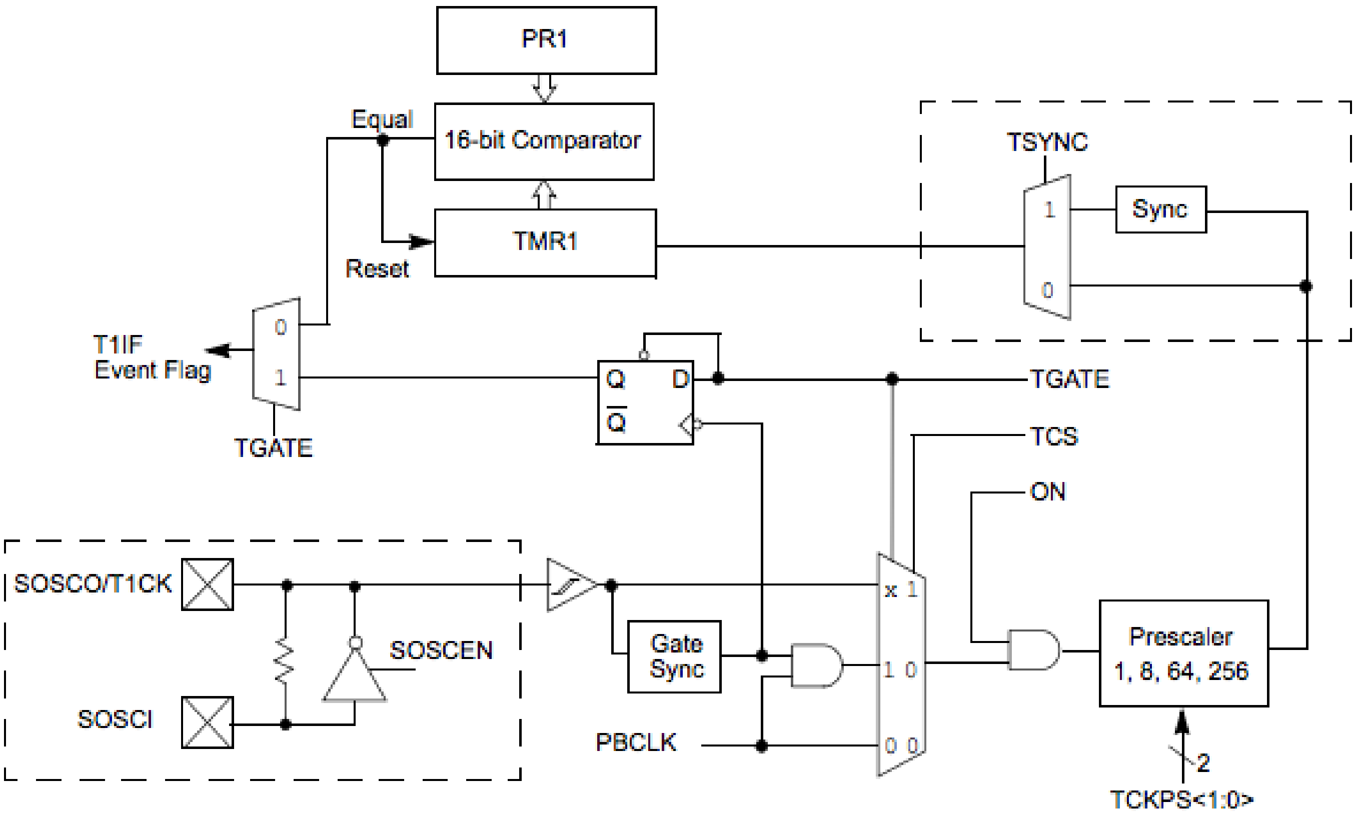
```
// Create specified ms/us of delay using built-in timer
#include <P32xxxx.h>

void delaymicros(int micros) {
  if (micros > 1000) {          // avoid timer overflow
    delaymicros(1000);
    delaymicros(micros-1000);
  }
  else if (micros > 6){
    TMR1 = 0;                   // reset timer to 0
    T1CONbits.ON = 1;           // turn timer on
    PR1 = (micros-6)*20;        // 20 clocks per microsecond
                                // Function has overhead of ~6 us
    IFS0bits.T1IF = 0;          // clear overflow flag
    while (!IFS0bits.T1IF);     // wait until overflow flag set
  }
}

void delaymillis(int millis) {
  while (millis--) delaymicros(1000); // repeatedly delay 1 ms
}                                     // until done
```

Timers

MEMORY & I/O SYSTEMS

- Timer1 is 16-bit timer
- 2^16-1 or 65,535
- SYSCLK = 40MHz,
- Prescalers
  - 1:1, 1:8, 1:64, and 1:256
  - Use T1CONbits.TCKPS=3 for 1:256 scaling
- Prescalar: 1:256, with 40MHz, the timer will be reset every 1/40e6 * 256 * 65535 = 0.419s.

Timer Implementation



Analog Input and Output

- Interface with the real world
- Analog--to--digital--converter(ADC)
- Digital--to--analog--converter(DAC)

Analog-to-Digital Conversion

- **Analog:** continuously valued signal, such as temperature or speed, with infinite possible values in between
- **Digital:** discretely valued signal, such as integers, encoded in binary
- Analog-to-digital converter: ADC, A/D, A2D; converts an analog signal to a digital signal
- Digital-to-analog converter: DAC, D/A, D2A
- An embedded system's surroundings typically involve many analog signals.

Analog Signals

- Analog signals – directly measurable quantities in terms of some other quantity
- Examples:
  - Thermometer – mercury height rises as temperature rises
  - Car Speedometer – Needle moves farther right as you accelerate
  - Stereo – Volume increases as you turn the knob.

MEMORY & I/O SYSTEMS

Digital Signals

- Digital Signals – have only two states. For digital computers, we refer to binary states, 0 and 1. “1” can be on, “0” can be off.
- Examples:
  - Light switch can be either on or off
  - Door to a room is either open or closed

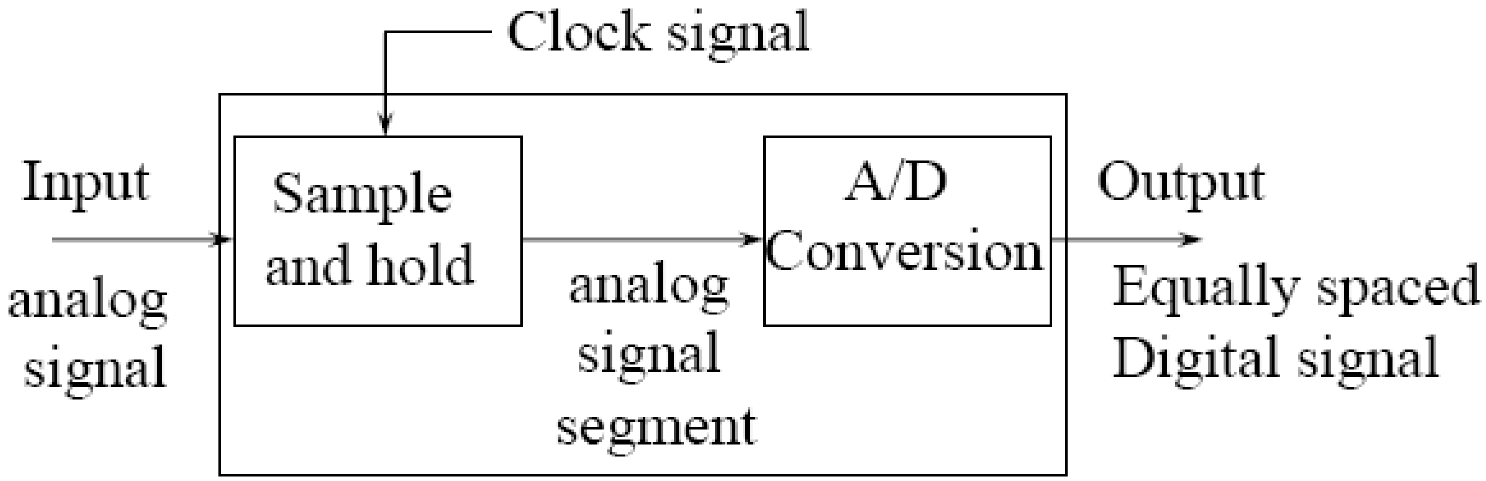
MEMORY & I/O SYSTEMS

Analog I/O

- Needed to interface with outside world
- **Analog input:** Analog-to-digital (A/D) conversion
  - Often included in microcontroller
  - *N*-bit: converts analog input from V_{ref-}-V_{ref+} to 0-2^{N-1}
- **Analog output:**
  - Digital-to-analog (D/A) conversion
    - Typically need external chip (e.g., AD558 or LTC1257)
    - *N*-bit: converts digital signal from 0-2^{N-1} to V_{ref-}-V_{ref+}
  - Pulse-width modulation

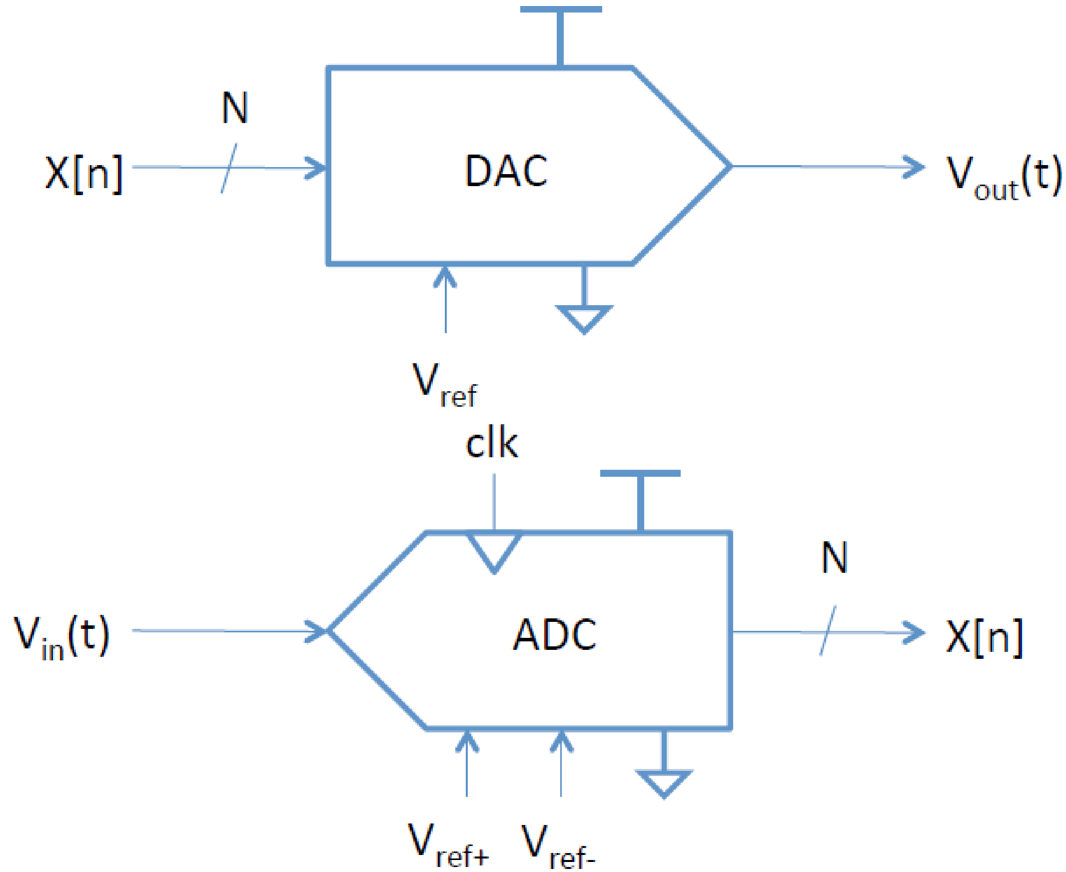
What does an A/D converter DO?

- Converts analog signals into binary words



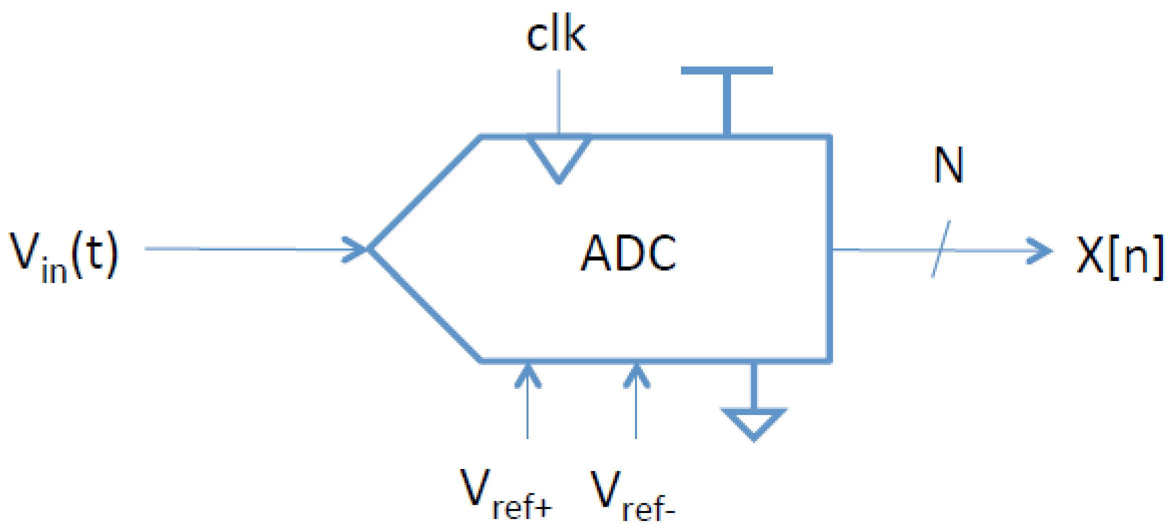
DAC/ADC Characterisc

- Resolution
- Dynamic range
- Sampling rate
- Accuracy

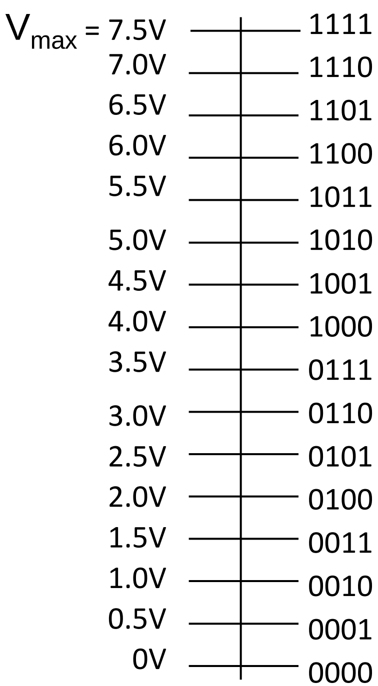


Example ADC

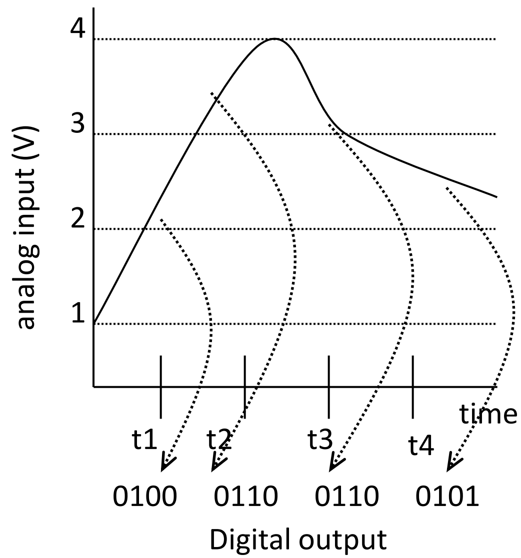
- Resolution: N = 12--bit
- Range: Vref- to Vref+ = 0-5 V
- Sampling fs = 44 KHz
- Accuracy: ± 3 least significant bits (lsbs)



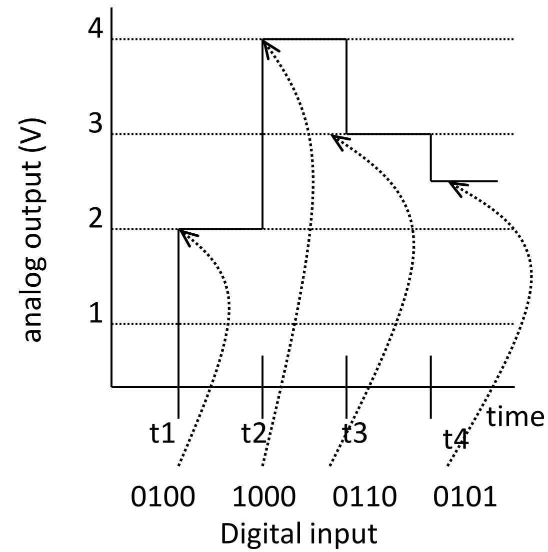
Analog-to-digital converters



proportionality



analog to digital



digital to analog

Proportional Signals

<u>Simple Equation</u>

Assume minimum voltage of 0 V.

Vmax = maximum voltage of the analog signal

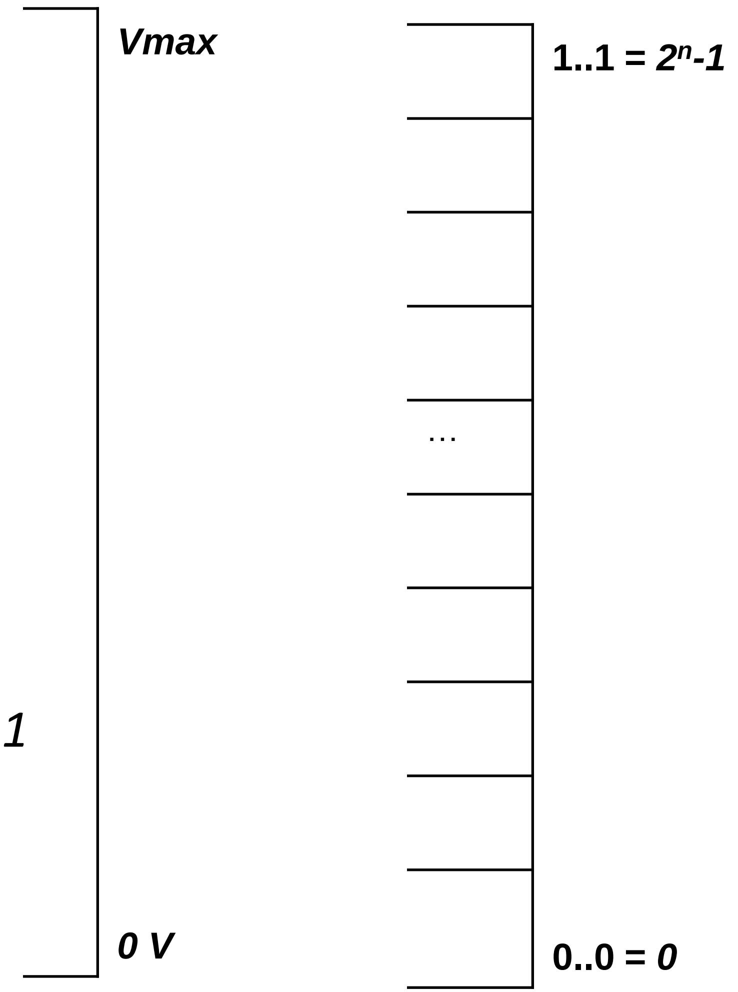
a = analog value

n = number of bits for digital encoding

- 2^n = number of digital codes
- ***M*** = number of steps, either 2^n or $2^n - 1$

d = digital encoding

$$a / Vmax = d / M$$



Resolution

Let $n = 2$

$\underline{M = 2^n - 1}$

3 steps on the digital scale

$d_0 = 0 = 0b00$

$d_{Vmax} = 3 = 0b11$

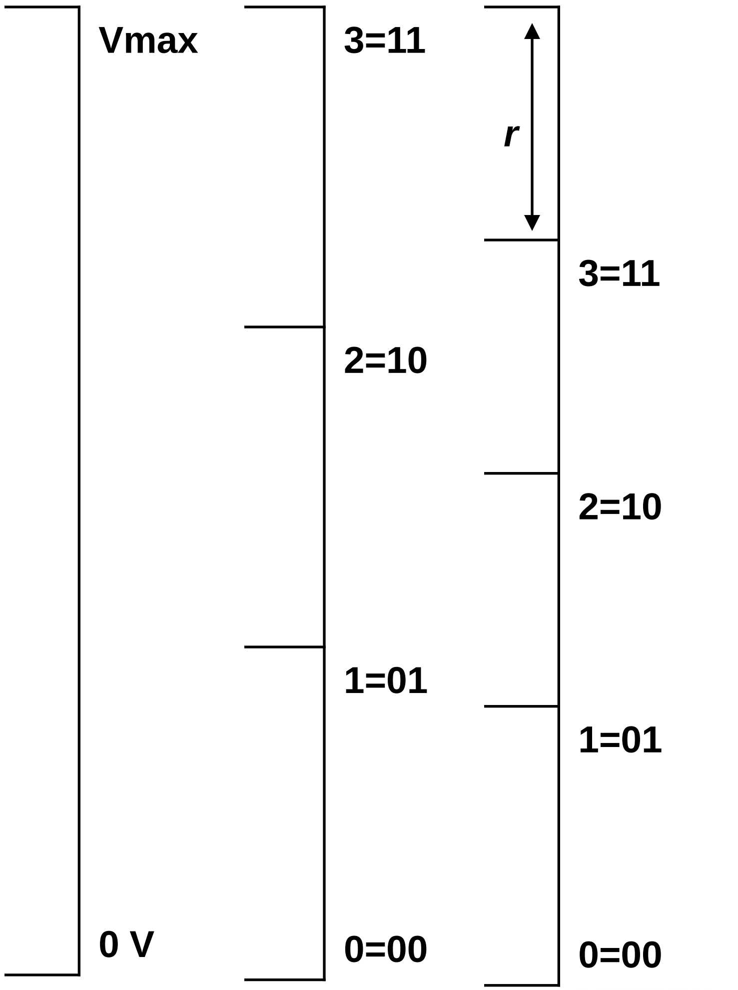
$\underline{M = 2^n}$

4 steps on the digital scale

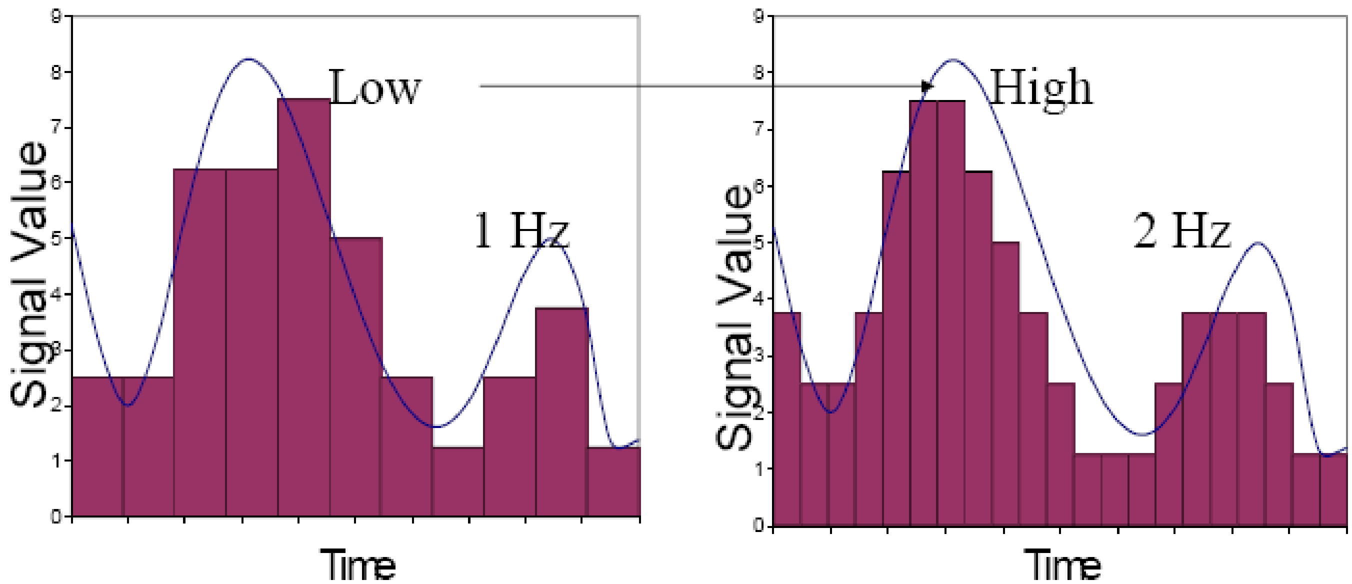
$d_0 = 0 = 0b00$

$d_{Vmax - r} = 3 = 0b11$ (no d_{Vmax})

***r*, resolution**: smallest analog change resulting from changing one bit



Sampling Rate



Frequency at which ADC evaluates analog signal. As we see in the second picture, evaluating the signal more often more accurately depicts the ADC signal.

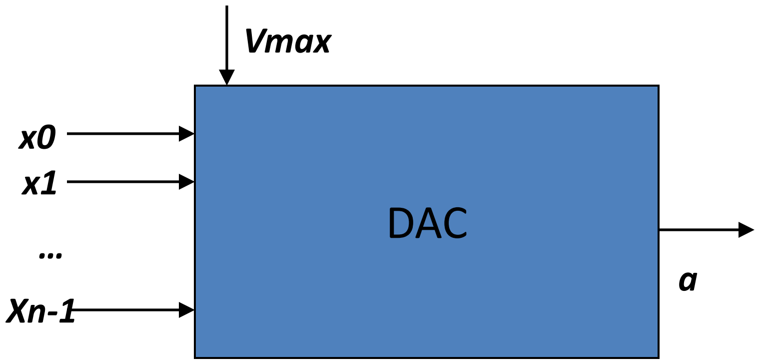
DAC Conversion

- No built-in DACs
- Some accept N-parallel wires
- Some accept serial (such as SPI)
- Flexible voltage vs. not
- May need an op-amp

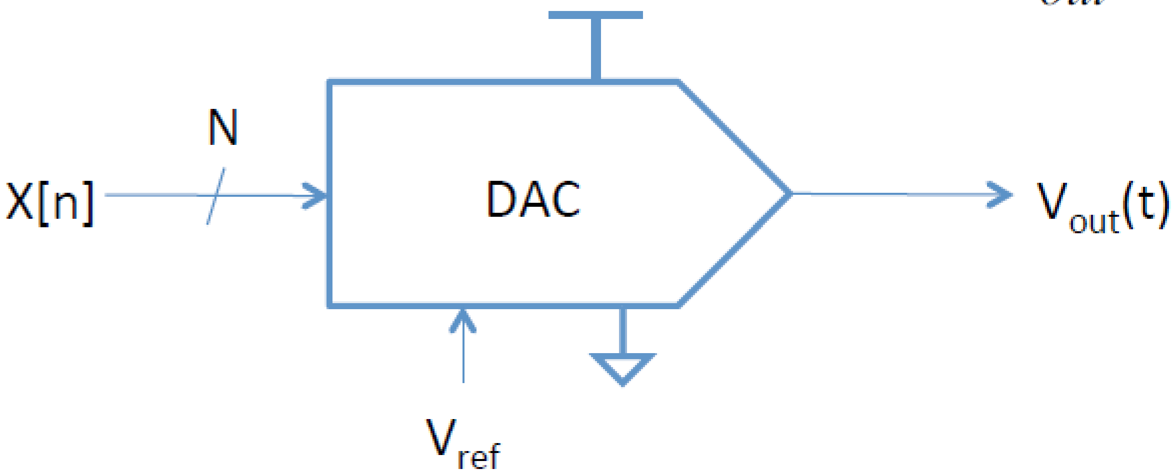
DAC vs. ADC

DAC:

n digital inputs for digital encoding *d*
analog input for *Vmax*
analog output *a*



$$V_{out}(t) = \frac{X[n]}{2^N} V_{ref}$$



Other Microcontroller Peripherals

- Examples
  - Character LCD
  - VGA monitor
  - Bluetooth wireless
  - Motors

Personal Computer (PC) I/O Systems

- USB: Universal Serial Bus
  - USB 1.0 released in 1996
  - standardized cables/software for peripherals
- PCI/PCIe: Peripheral Component Interconnect/PCI Express
  - developed by Intel, widespread around 1994
  - 32-bit parallel bus
  - used for expansion cards (i.e., sound cards, video cards, etc.)
- DDR: double-data rate memory

Personal Computer (PC) I/O Systems

- TCP/IP: Transmission Control Protocol and Internet Protocol
  - physical connection: Ethernet cable or Wi-Fi
- SATA: hard drive interface
- Input/Output (sensors, actuators, microcontrollers, etc.)
  - Data Acquisition Systems (DAQs)
  - USB Links