Microprocessor – Microcontroller

201 W01 – **Microcontroller Basics**

Nguyen Tran Huu Nguyen

D: Computer Engineering

E: nthnguyen@hcmut.edu.vn

Differences between Microprocessors and Microcontrollers

- Microprocessors (µPs)
  - General-purpose compute "engine"
  - External memory and I/O devices
  - Often requires an operating system (OS)

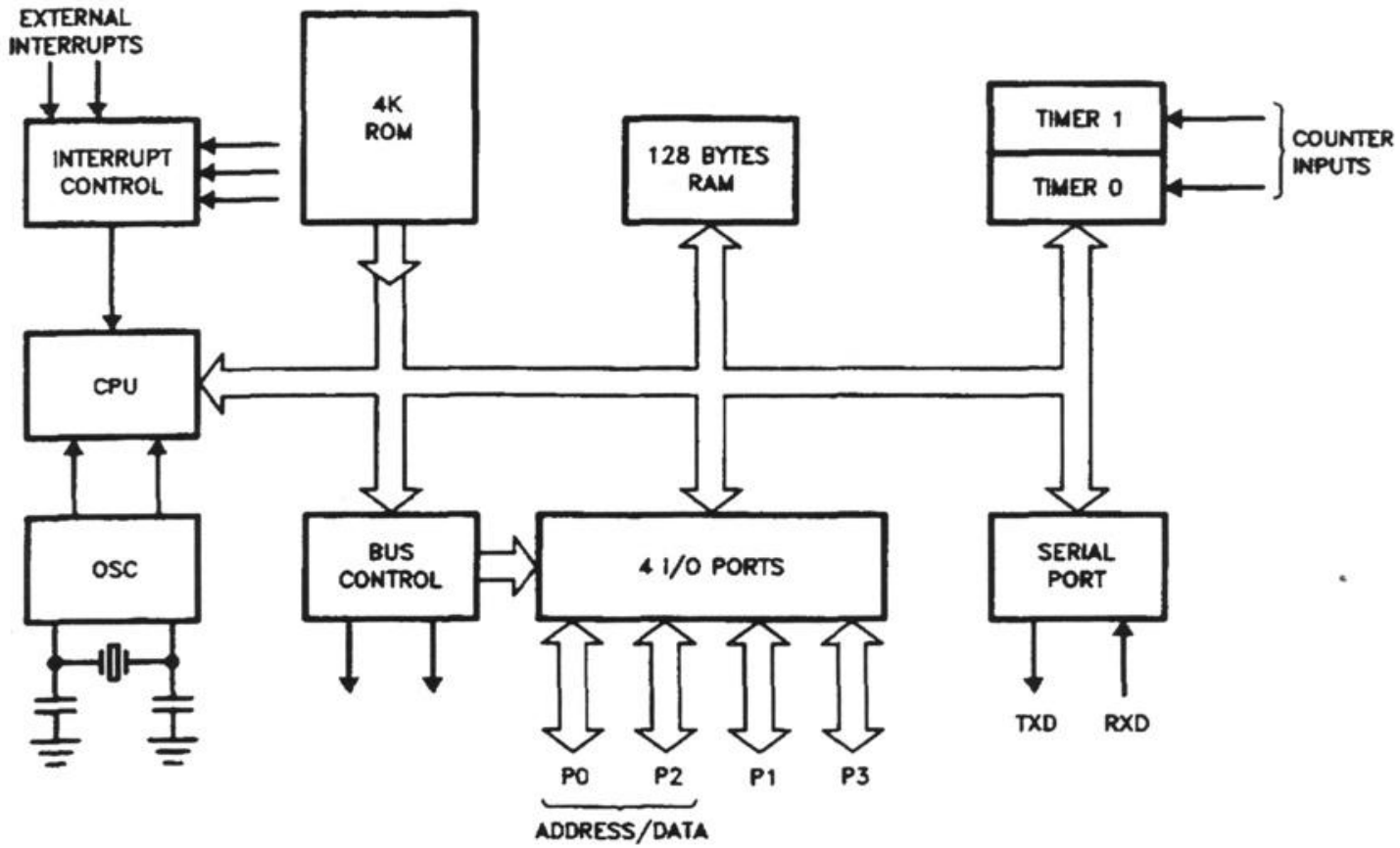
Differences between Microprocessors and Microcontrollers

- Microcontrollers (MCUs)
  - Usually chosen for a specific purpose
  - Small packages
  - On-chip memory and peripherals
  - Fast "on time," no BIOS or OS needed

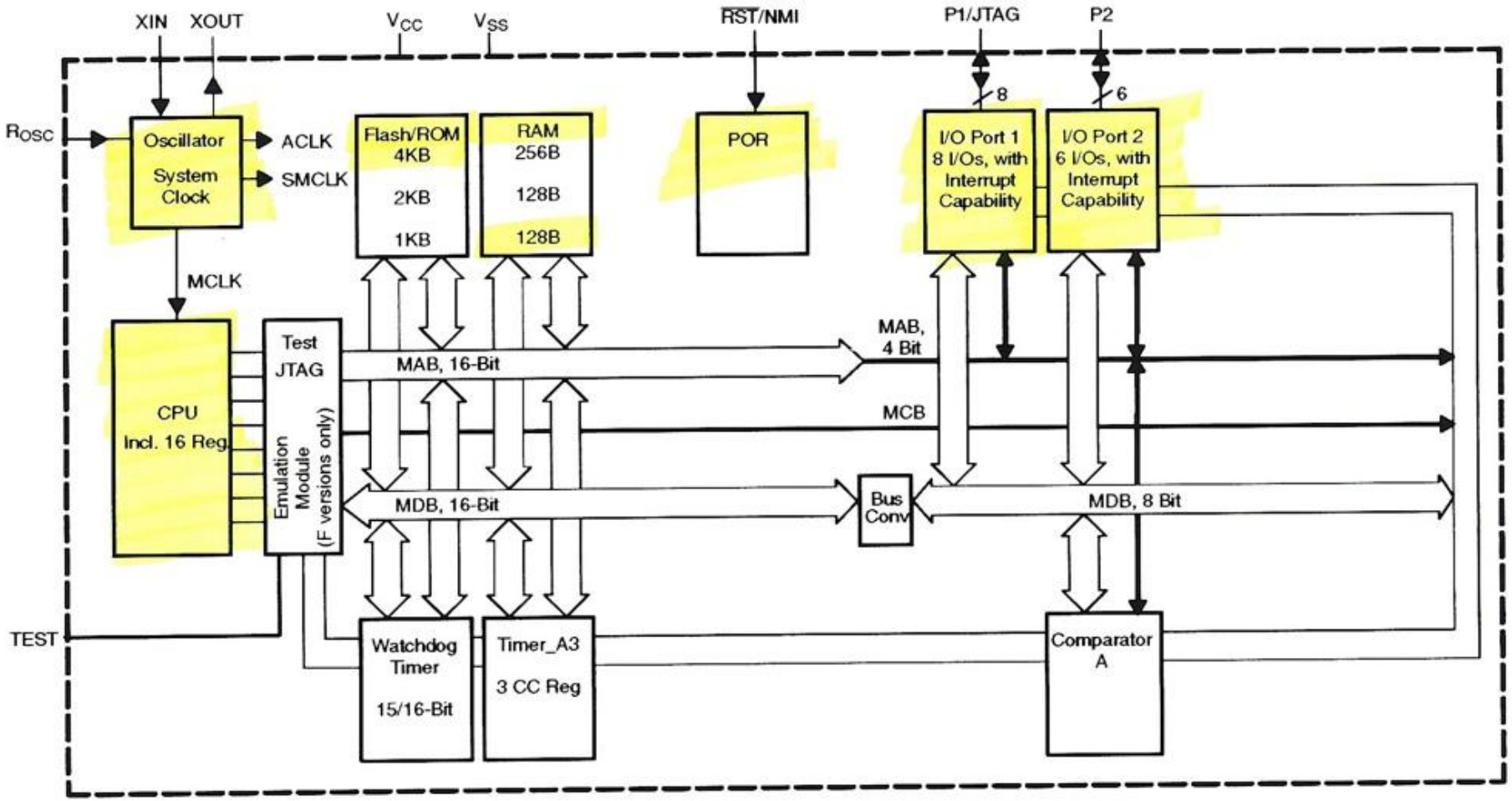
Where Did the MCU Come From?

- Intel introduced the 8051 MCU in 1980
  - Small amount of read-only memory (ROM)
  - External memory expansion if needed
  - Four 8-bit I/O ports
  - Not much different from today's MCUs

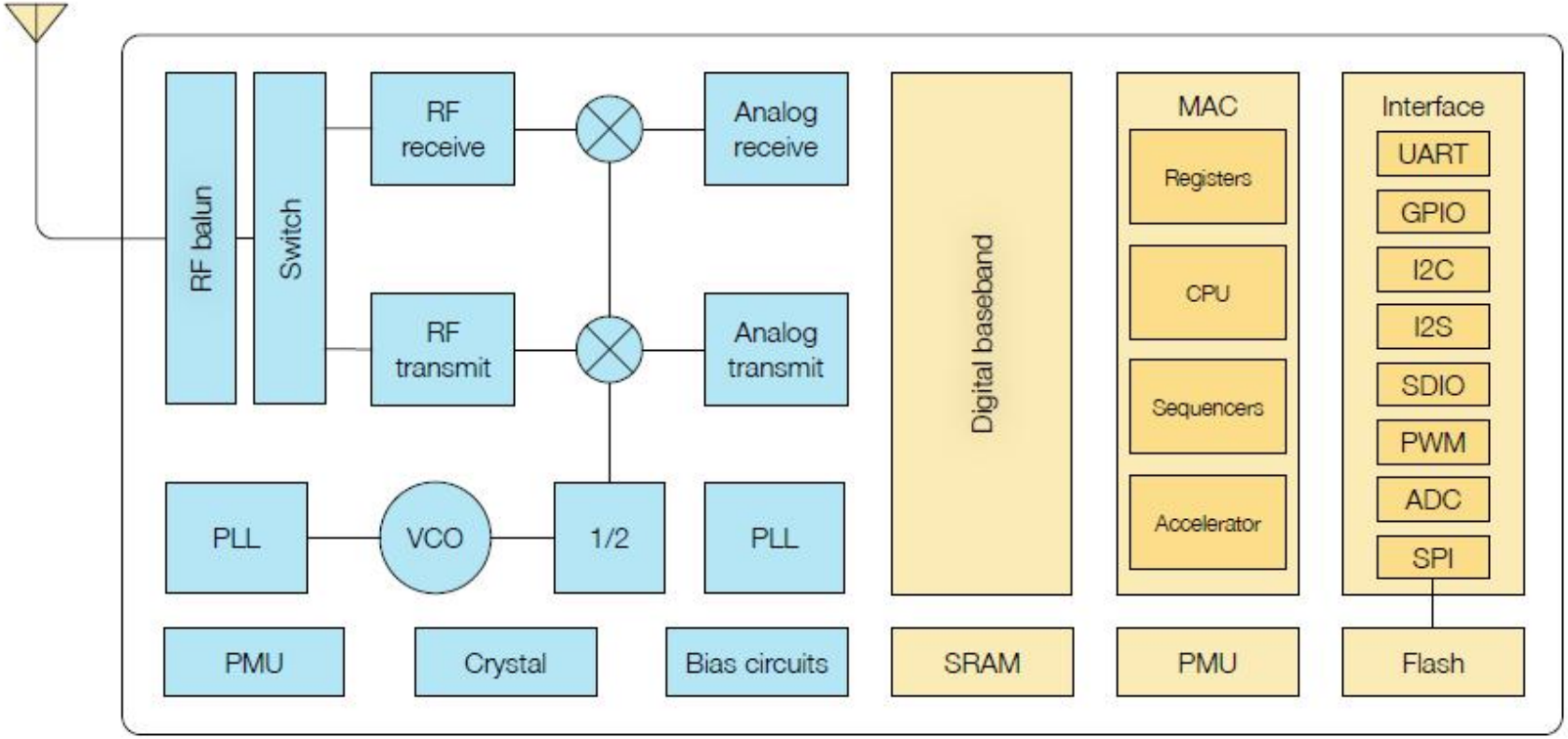
8051 Architecture (1980)



Texas Instruments MSP430 MCU



ESP8266


RF balun
Switch
RF receive
RF transmit
Analog receive
Analog transmit
PLL
VCO
1/2
PLL
Digital baseband
MAC
Registers
CPU
Sequencers
Accelerator
Interface
UART
GPIO
I2C
I2S
SDIO
PWM
ADC
SPI
PMU
Crystal
Bias circuits
SRAM
PMU
Flash

PIC18F8722

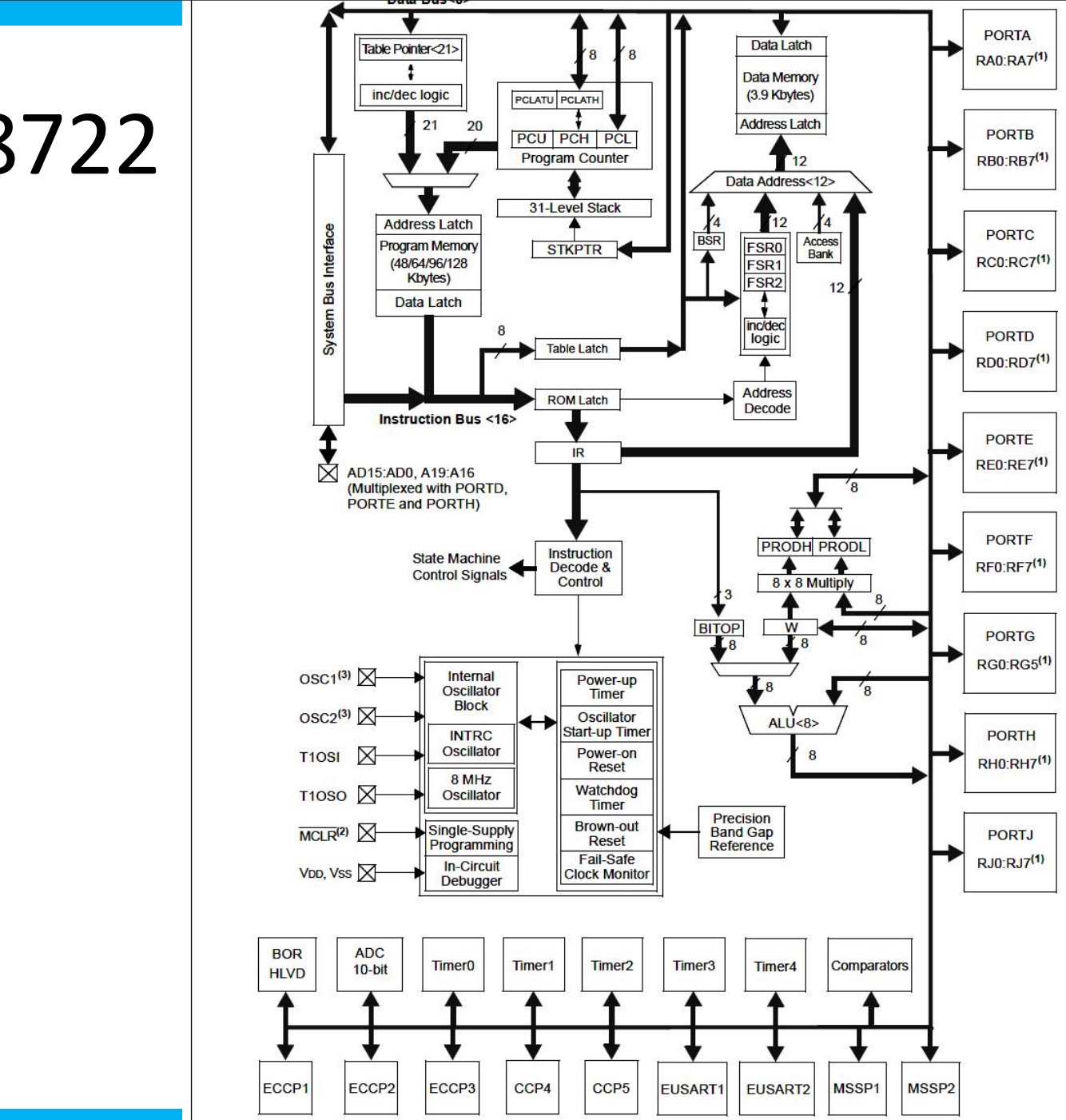
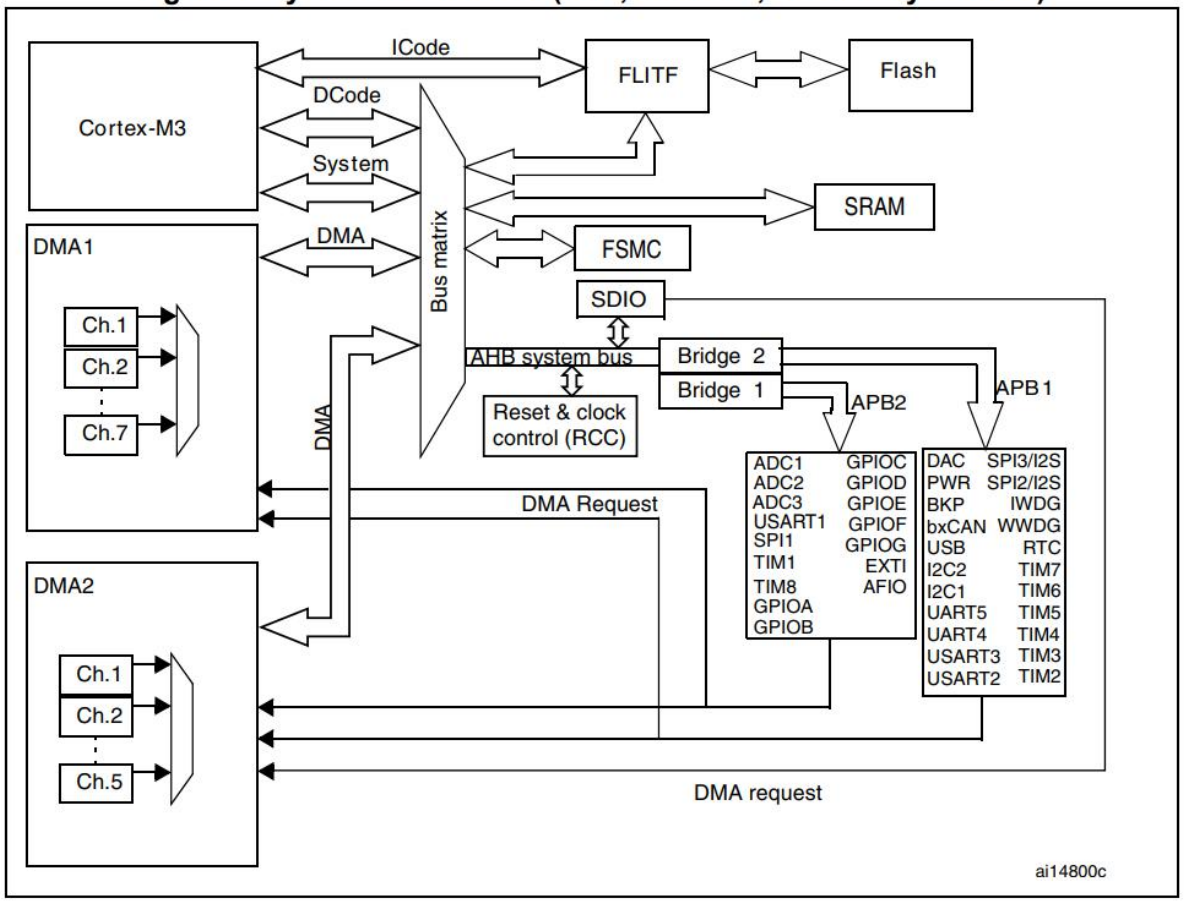


Table Pointer<21>
inc/dec logic
21
20
8
8
PCLATU
PCLATH
PCU
PCH
PCL
Program Counter
31-Level Stack
STKPTR
Address Latch
Program Memory (48/64/96/128 Kbytes)
Data Latch
System Bus Interface
Table Latch
ROM Latch
Instruction Bus <16>
IR
AD15:AD0, A19:A16 (Multiplexed with PORTD, PORTE and PORTH)
Data Latch
Data Memory (3.9 Kbytes)
Address Latch
12
Data Address<12>
4
BSR
12
FSR0
FSR1
FSR2
inc/dec logic
4
Access Bank
12
Address Decode
State Machine Control Signals
Instruction Decode & Control
PRODH
PRODL
8 x 8 Multiply
3
BITOP
W
8
ALU<8>
OSC1(3)
OSC2(3)
T1OSI
T1OSO
MCLR(2)
VDD, VSS
Internal Oscillator Block
INTRC Oscillator
8 MHz Oscillator
Single-Supply Programming
In-Circuit Debugger
Power-up Timer
Oscillator Start-up Timer
Power-on Reset
Watchdog Timer
Brown-out Reset
Fail-Safe Clock Monitor
Precision Band Gap Reference
PORTA RA0:RA7(1)
PORTB RB0:RB7(1)
PORTC RC0:RC7(1)
PORTD RD0:RD7(1)
PORTE RE0:RE7(1)
PORTF RF0:RF7(1)
PORTG RG0:RG5(1)
PORTH RH0:RH7(1)
PORTJ RJ0:RJ7(1)
BOR HLVD
ADC 10-bit
Timer0
Timer1
Timer2
Timer3
Timer4
Comparators
ECCP1
ECCP2
ECCP3
CCP4
CCP5
EUSART1
EUSART2
MSSP1
MSSP2

STM32

Figure 1. System architecture (low-, medium-, XL-density devices)



Why Use an MCU?

- Everything in one small package
- Mix and match peripherals and I/O types
- Lots of memory, flash for code, SRAM for data
- Readily available hardware and software tools
- Helpful support communities and forums
- Reference designs
- Code libraries and examples

What Peripherals Do MCUs Offer?

- Digital I/O -- On or Off
  - Parallel signals
  - Pulse-width-modulated logic signals
  - Counters and timers

- Analog I/O -- Voltages
  - Comparators
  - Analog-to-digital converters (ADCs)
  - Digital-to-analog converters (DACs)

What Peripherals Do MCUs Offer?

- Communication Devices
  - UART or USART
  - SPI
  - I2C
  - I2S
  - CAN
  - USB
  - Ethernet
- Interrupts

Peripherals Devices in MCUs

- Parallel I/O Ports
  - Usually 8 or 16 bits for simultaneous control
  - Toggle individual bits
  - Require setup of registers

- Parallel I/O-Port Examples

How Do I Set Up I/O Ports?

TABLE 2-1: PIC16F631/677/685/687/689/690 SPECIAL FUNCTION REGISTERS SUMMARY BANK 0

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Page
Bank 0											
00h	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	44,205
01h	TMR0	Timer0 Module Register								xxxx xxxx	81,205
02h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000	44,205
03h	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	36,205
04h	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	44,205
05h	PORTA(7)	—	—	RA5	RA4	RA3	RA2	RA1	RA0	--xx xxxx	59,205
06h	PORTB(7)	RB7	RB6	RB5	RB4	—	—	—	—	xxxx ----	69,205
07h	PORTC(7)	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx	76,205
08h	—	Unimplemented								—	—
09h	—	Unimplemented								—	—
0Ah	PCLATH	—	—	—	Write Buffer for upper 5 bits of Program Counter					---0 0000	44,205
0Bh	INTCON	GIE	PEIE	T0IE	INTE	RABIE	T0IF	INTF	RABIF(1)	0000 000x	38,205
0Ch	PIR1	—	ADIF(4)	RCIF(2)	TXIF(2)	SSPIF(5)	CCP1IF(3)	TMR2IF(3)	TMR1IF	-000 0000	41,205
0Dh	PIR2	OSFIF	C2IF	C1IF	EEIF	—	—	—	—	0000 ----	42,205
0Eh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	86,205
0Fh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	86,205
10h	T1CON	T1GINV	TMR1GE	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	0000 0000	88,205
11h	TMR2(3)	Timer2 Module Register								0000 0000	91,205

Table 59. GPIO register map and reset values

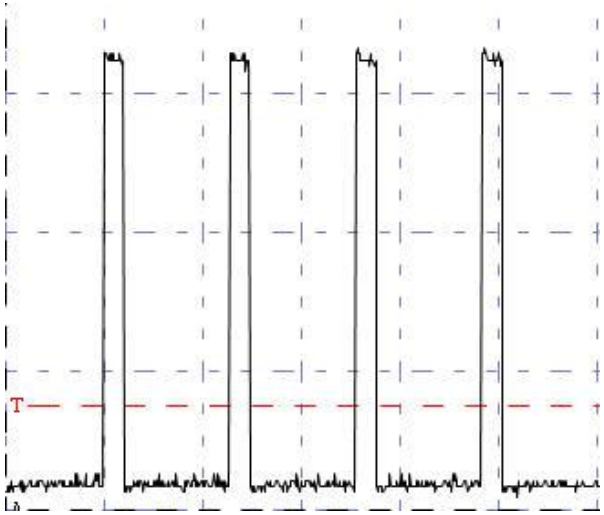
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	GPIOx_CRL	CNF7[1:0]		MODE7[1:0]		CNF6[1:0]		MODE6[1:0]		CNF5[1:0]		MODE5[1:0]		CNF4[1:0]		MODE4[1:0]		CNF3[1:0]		MODE3[1:0]		CNF2[1:0]		MODE2[1:0]		CNF1[1:0]		MODE1[1:0]		CNF0[1:0]		MODE0[1:0]	
	Reset value	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0
0x04	GPIOx_CRH	CNF15[1:0]		MODE15[1:0]		CNF14[1:0]		MODE14[1:0]		CNF13[1:0]		MODE13[1:0]		CNF12[1:0]		MODE12[1:0]		CNF11[1:0]		MODE11[1:0]		CNF10[1:0]		MODE10[1:0]		CNF9[1:0]		MODE9[1:0]		CNF8[1:0]		MODE8[1:0]	
	Reset value	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0
0x08	GPIOx_IDR	Reserved																IDRy															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	GPIOx_ODR	Reserved																ODRy															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	GPIOx_BSRR	BR[15:0]																BSR[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	GPIOx_BRR	Reserved																BR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	GPIOx_LCKR	Reserved															LCKK	LCK[15:0]															
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

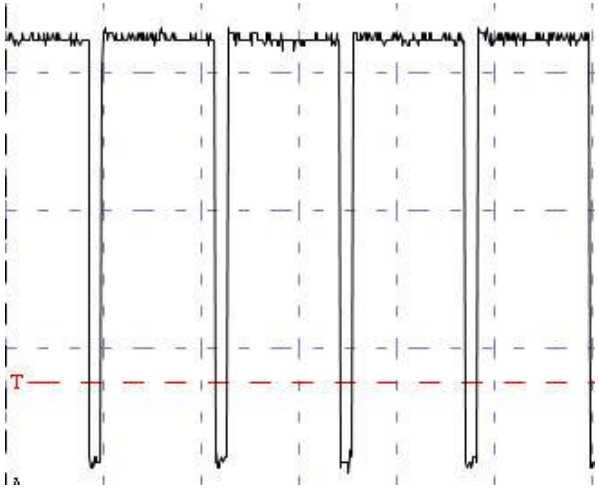
Table 60. AFIO register map and reset values

Offset	Register																															
0x00	AFIO_EVCR	Reset value																													EVOE	PORT[2:0]		PIN[3:0]	
																															0	0	0	0	0
0x04	AFIO_MAPR low-, medium-, high- and XL-density devices	Reset value	Reserved																															
			Reserved																															
			SWJ_CFG[2:0]																															
			0	0	0																													
			Reserved																															
			0	0	0																													
			ADC2_ETRGREG_REMAP	ADC2_ETRGINJ_REMAP	ADC1_ETRGREG_REMAP	ADC1_ETRGINJ_REMAP	TIM5CH4_IREMAP	PD01_REMAP	CAN1_REMAP[1]	CAN1_REMAP[0]	TIM4_REMPAP	TIM3_REMPAP[1]	TIM3_REMPAP[0]	TIM2_REMPAP[1]	TIM2_REMPAP[0]	TIM1_REMPAP[1]	TIM1_REMPAP[0]	USART3_REMAP[1]	USART3_REMAP[0]	USART2_REMAP	USART1_REMAP	I2C1_REMAP	SPI1_REMAP										
0x04	AFIO_MAPR connectivity line devices	Reset value	Reserved																															
			Reserved																															
			SWJ_CFG[2:0]																															
			0	0	0																													
			Reserved																															
			0	0	0																													
			Reserved																															
			MII_RMII_SEL	CAN2_REMAP	ETH_REMAP	Reserved	TIM5CH4_IREMAP	PD01_REMAP	CAN1_REMAP[1]	CAN1_REMAP[0]	TIM4_REMAP	TIM3_REMAP[1:0]		TIM2_REMAP[1:0]		TIM1_REMAP[1:0]		USART3_REMAP[1:0]		USART2_REMAP	USART1_REMAP	I2C1_REMAP	SPI1_REMAP										

Pulse-Width Modulator

- PWM Peripheral
  - Converts a value to a proportional pulse width
  - Operate continuously and independently
  - Motor, LED, servo, and power control



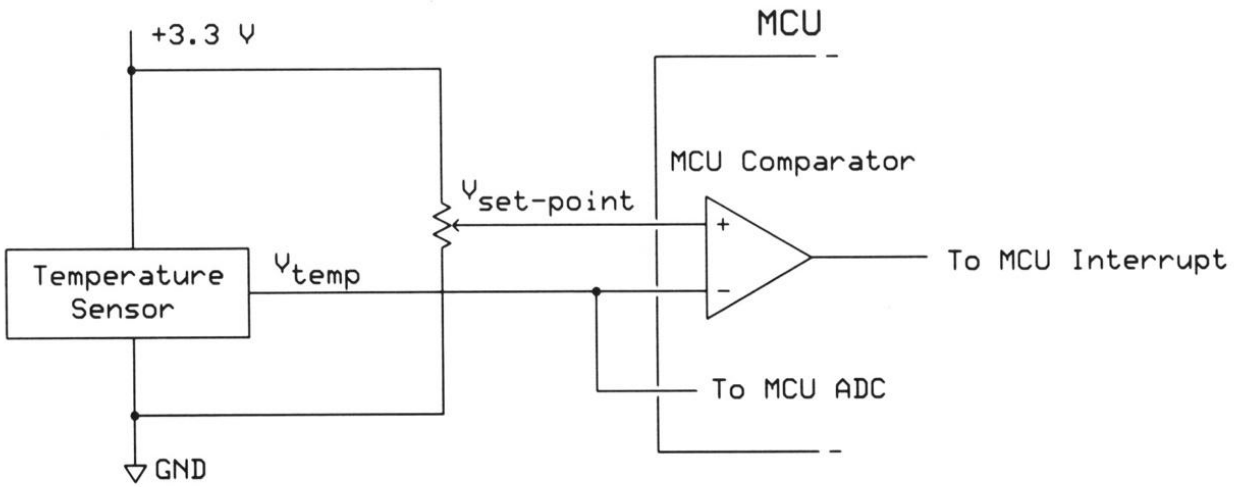


Counters and Timers

- Operate for a specific period or create a delay
- Count external events, count up or down
- Count clock "ticks" between the same or different events
- Choose from various clock sources

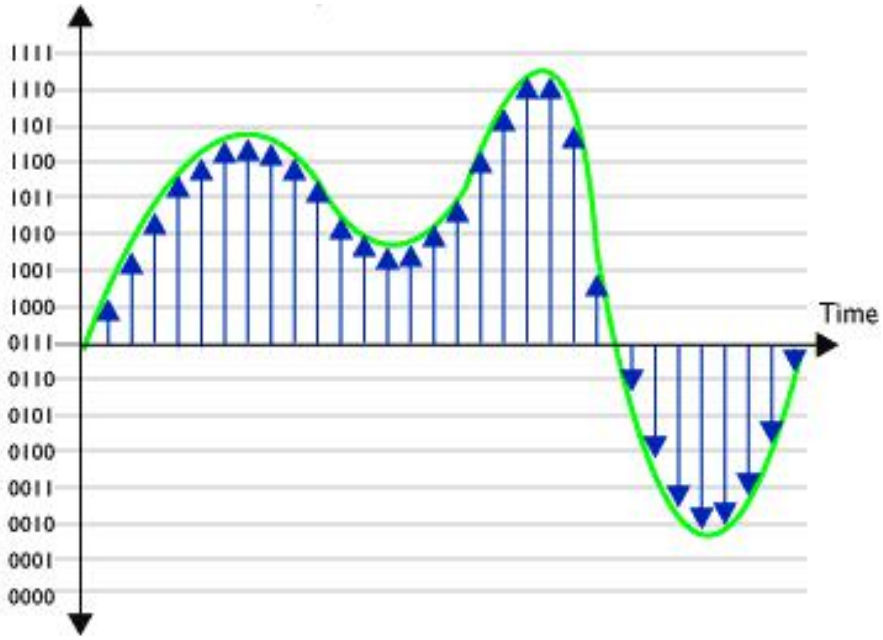
Analog Comparator

- Compare two voltages and...
  - Cause a bit to change state
  - Generate an interrupt
  - Wake an MCU from a sleep state



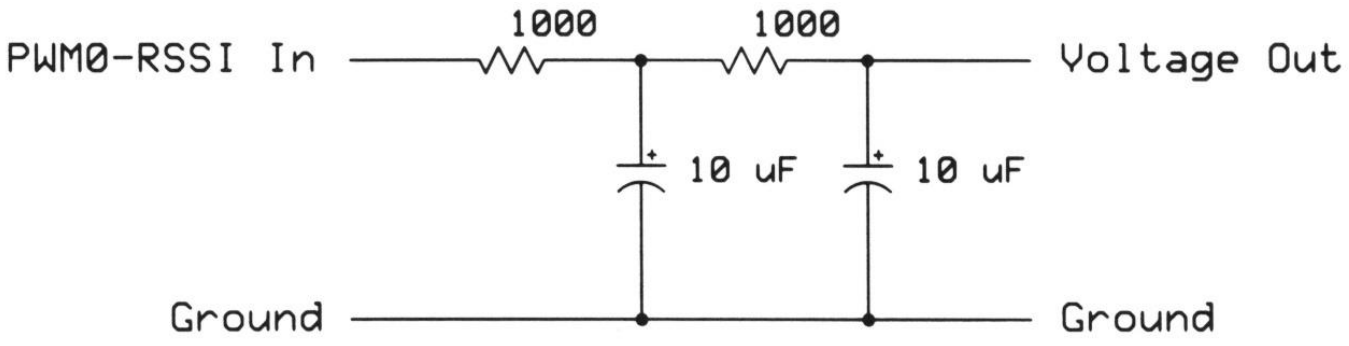
Analog-to-Digital Converter

- Convert a voltage to a digital value; 8, 10, 12 bits...
- Unipolar, 0 volts to MCU V+ (+3.3 or 5 volts)
- Might require external signal conditioning



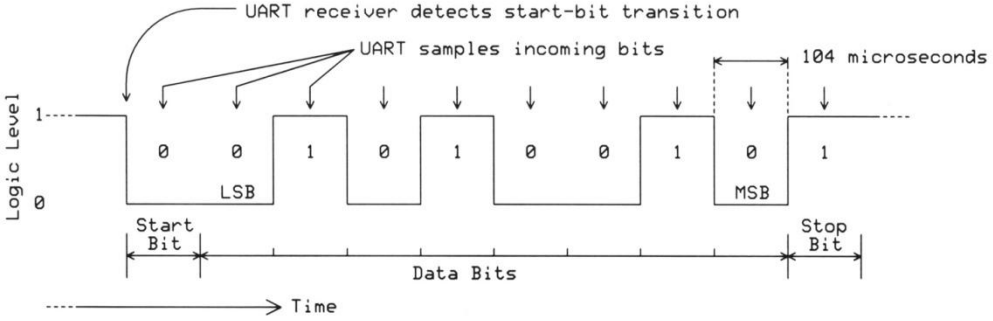
Digital-to-Analog Converter

- Unipolar output, might require external offset
- High-impedance output, could require buffering
- 10- and 12-bit DACs common on MCUs
- Filter a PWM output to get an analog voltage

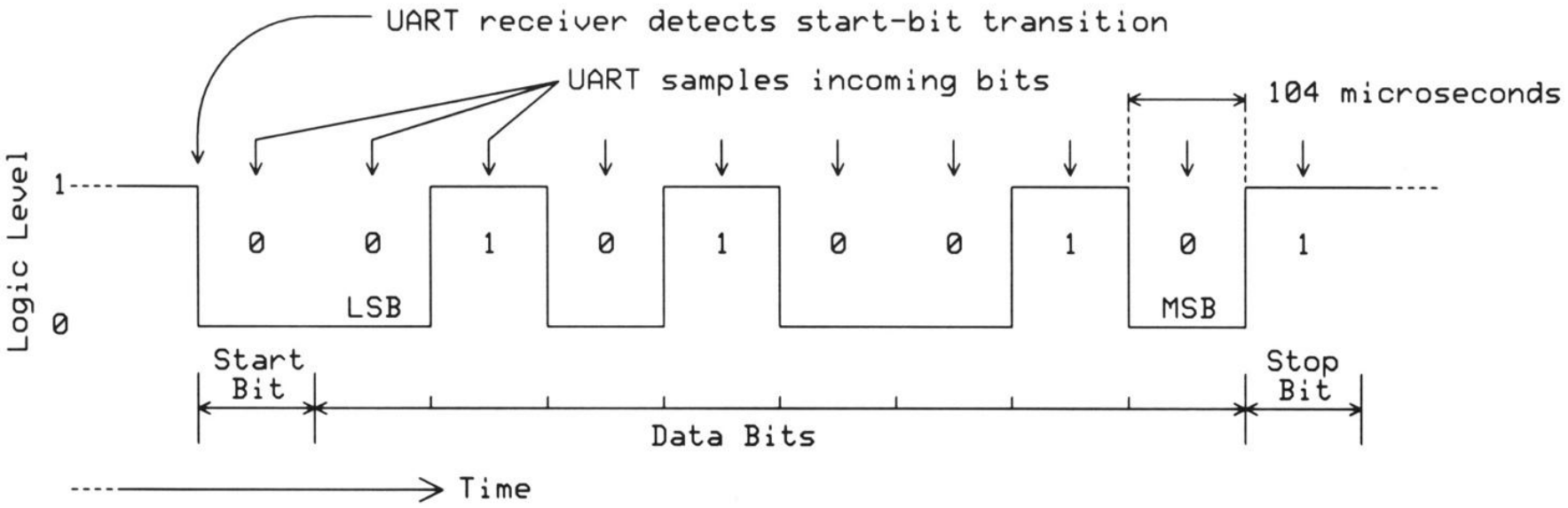


UART Communications

- Universal Asynchronous Receiver-Transmitter (UART)
  - Serial communications
  - Self-timing operations
  - Usually 8-bit transmissions at standard rates
  - Common on most MCUs



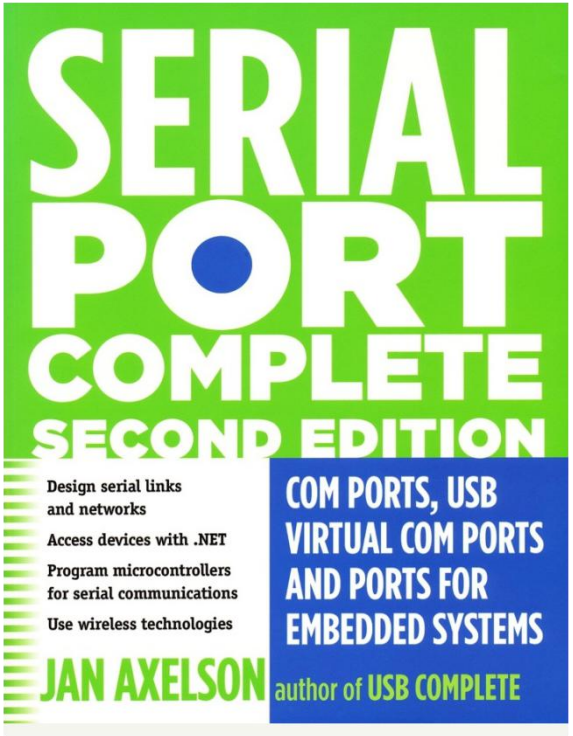
UART Communications Timing



Communications at 9600 bits/sec

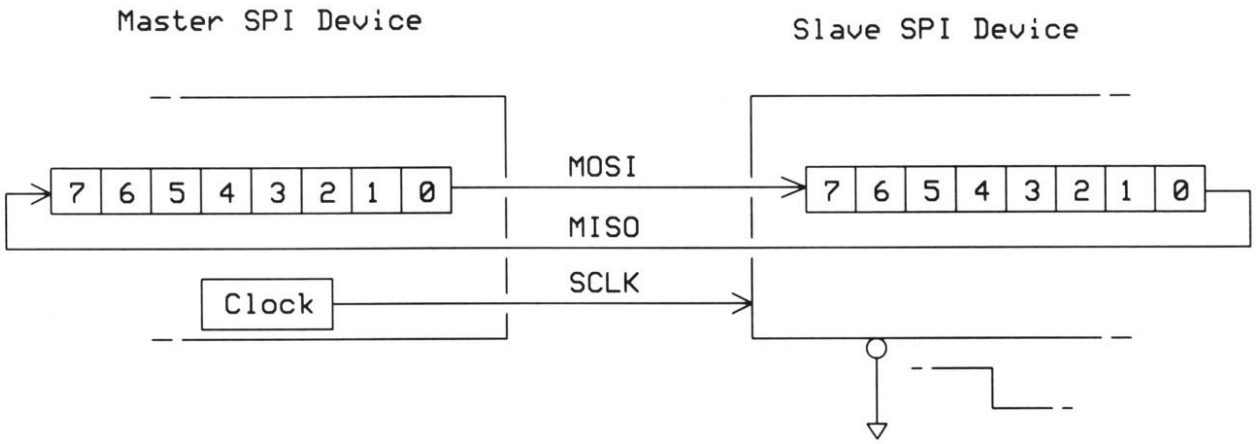
Serial-Communications

- Reference
  - “Serial Port Complete,” 2nd ed., by Jan Axelson, Lakeview Research, 2007. ISBN: 978-1-931448-06-2.



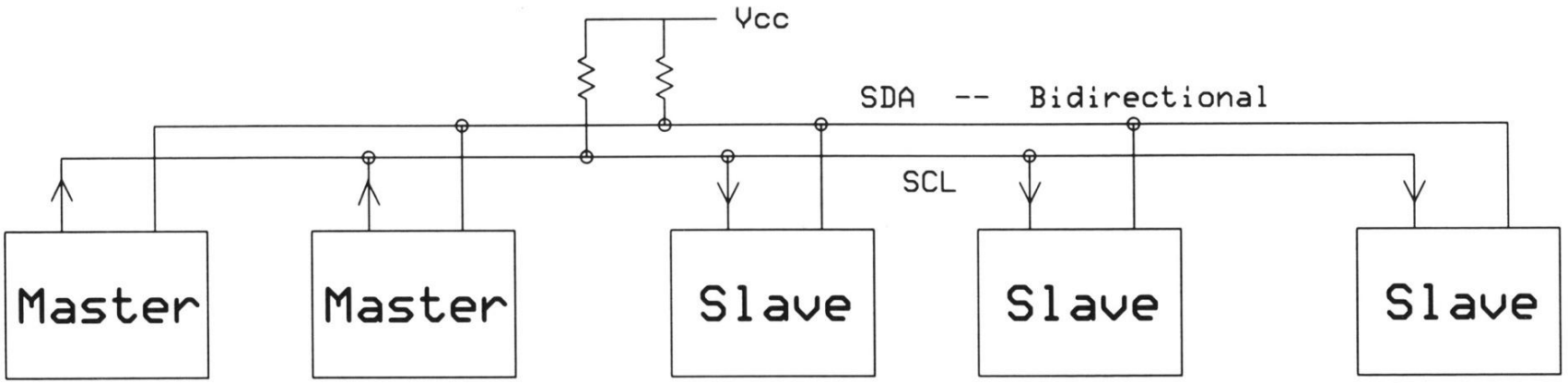
Serial Peripheral Interface (SPI)

- Used for chip-to-chip communications
- Requires a clock signal to all SPI ("spy") devices
- Not a formal standard
- Operates with ADCs, DACs, real-time clocks

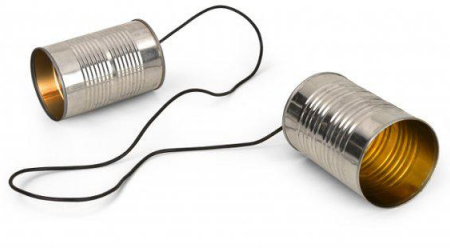


Inter Integrated-Circuit Interface (I^2C)

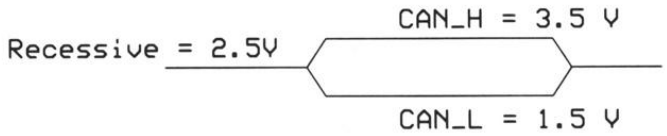
- Similar to SPI communications, but two wires
- Multiple masters and slaves
- Acknowledgements and bus arbitration
- Philips (NXP) standard (Rev. 3, June 2007)

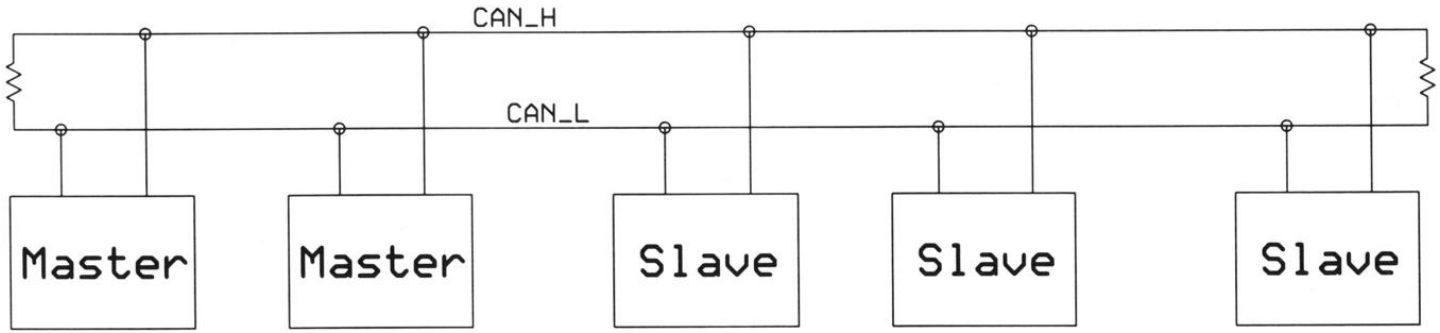


Controller-Area Network (CAN)



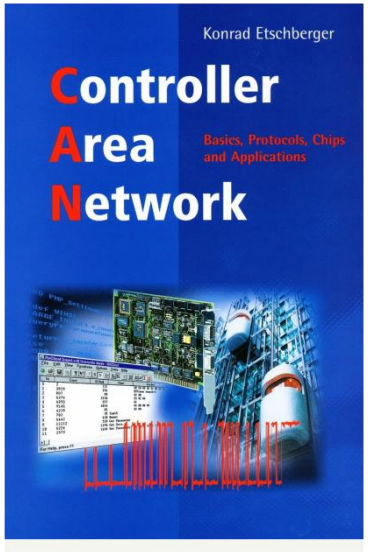
- A standard for vehicle equipment
- Uses an ISO-type "stack"
- 2-wire differential bus, no common ground needed
- Uses standardized packets of information

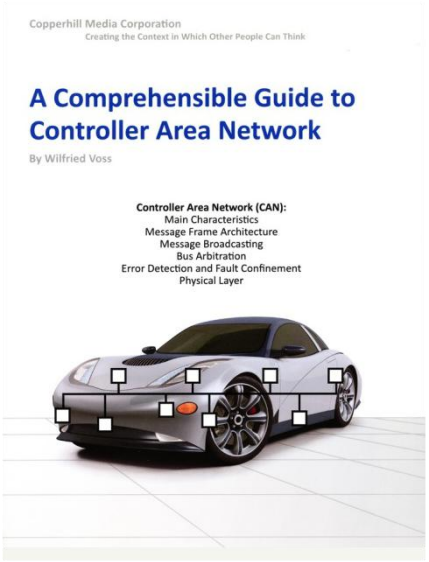




Controller-Area Network (CAN)

- References
  - ISO Standard 11898-x ($)
  - "Controller Area Network," by Konrad Etschberger," IXXAT Automation, 2001. ISB: 978-3-00-007376-0.
  - "A Comprehensive Guide to Controller Area Network," by Wilfried Voss, Copper Hill Media, 2008. ISBN: 978-0976511601.





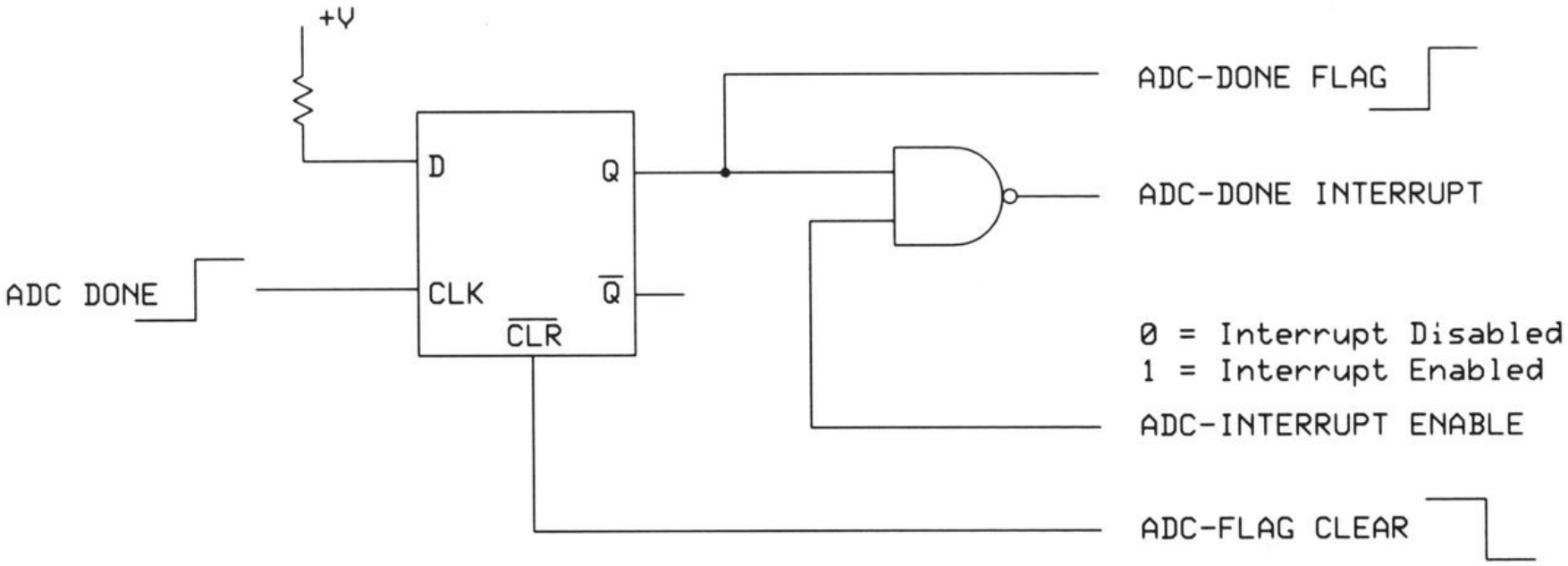
Ethernet and USB

- Governed by standards
- Require a software "stack"
  - Purchase, license, or create one yourself
  - MCU vendors might have stacks
- Some MCUs include everything except the physical interface (PHY)
- Can demand considerable memory
- Start with a development kit or reference design

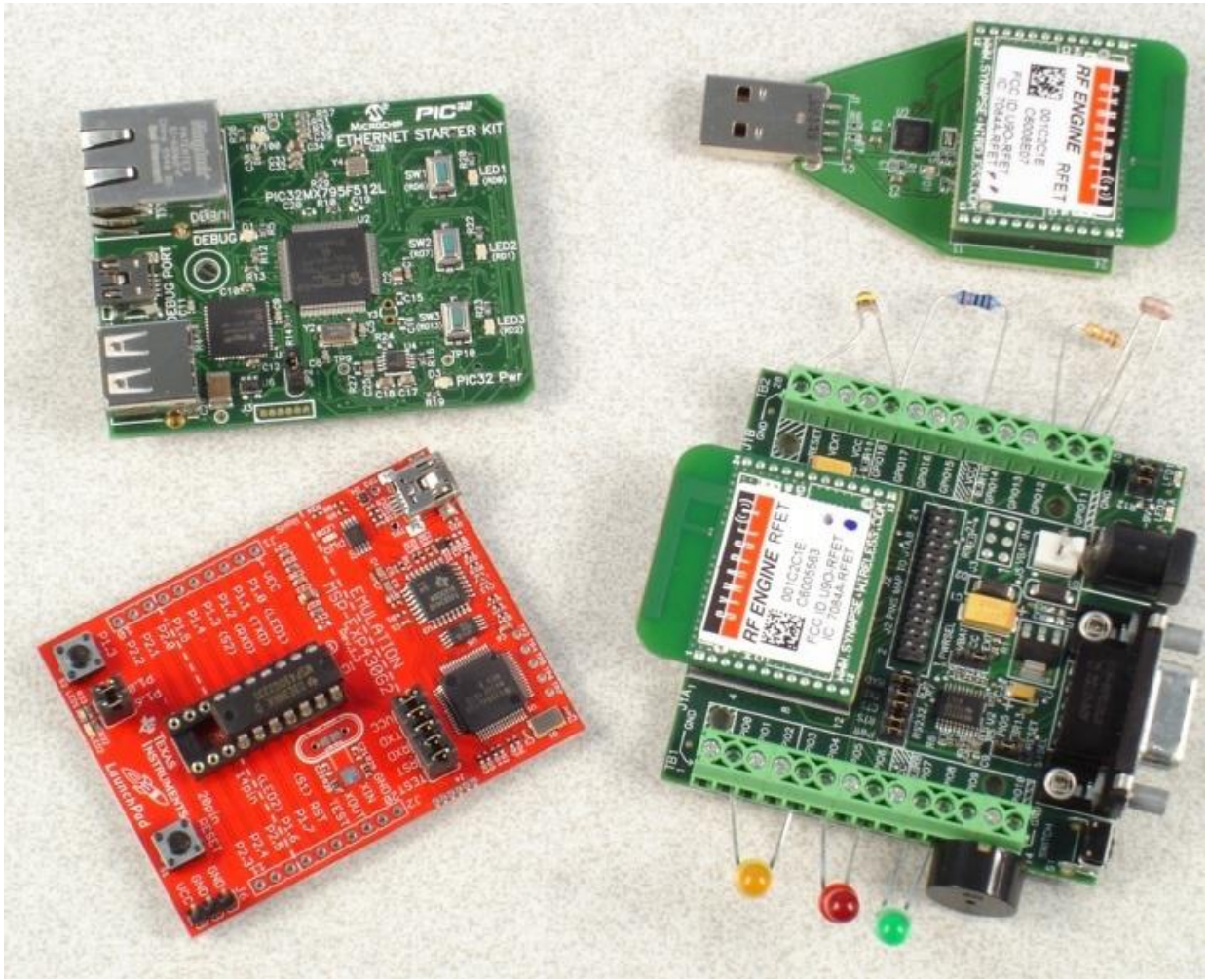
Interrupts

- Cause immediate action
- Internal and external hardware and software sources
- Two types of action -- one or many vectors
- Can present debug challenges

An ADC Interrupt



How Do I Get a Quick Start?



How Do I Get Started?