Chapter 1 – Introduction to Real-Time Embedded Systems

Overall Course Goal

To prepare the student to be able to start contributing

in one or more of the following areas when they get a job:

"Specify, design, implement, test, and/or manufacture real-time embedded systems."

Quotes for the day:

"It has been estimated that 99% of the world-wide production of microprocessors is used in embedded systems." Burns and Wellings, 2009

"Real-time embedded design has many facets." DNR 2017

Basic Definition of Real-Time Embedded System:

Define "***Embedded System***"

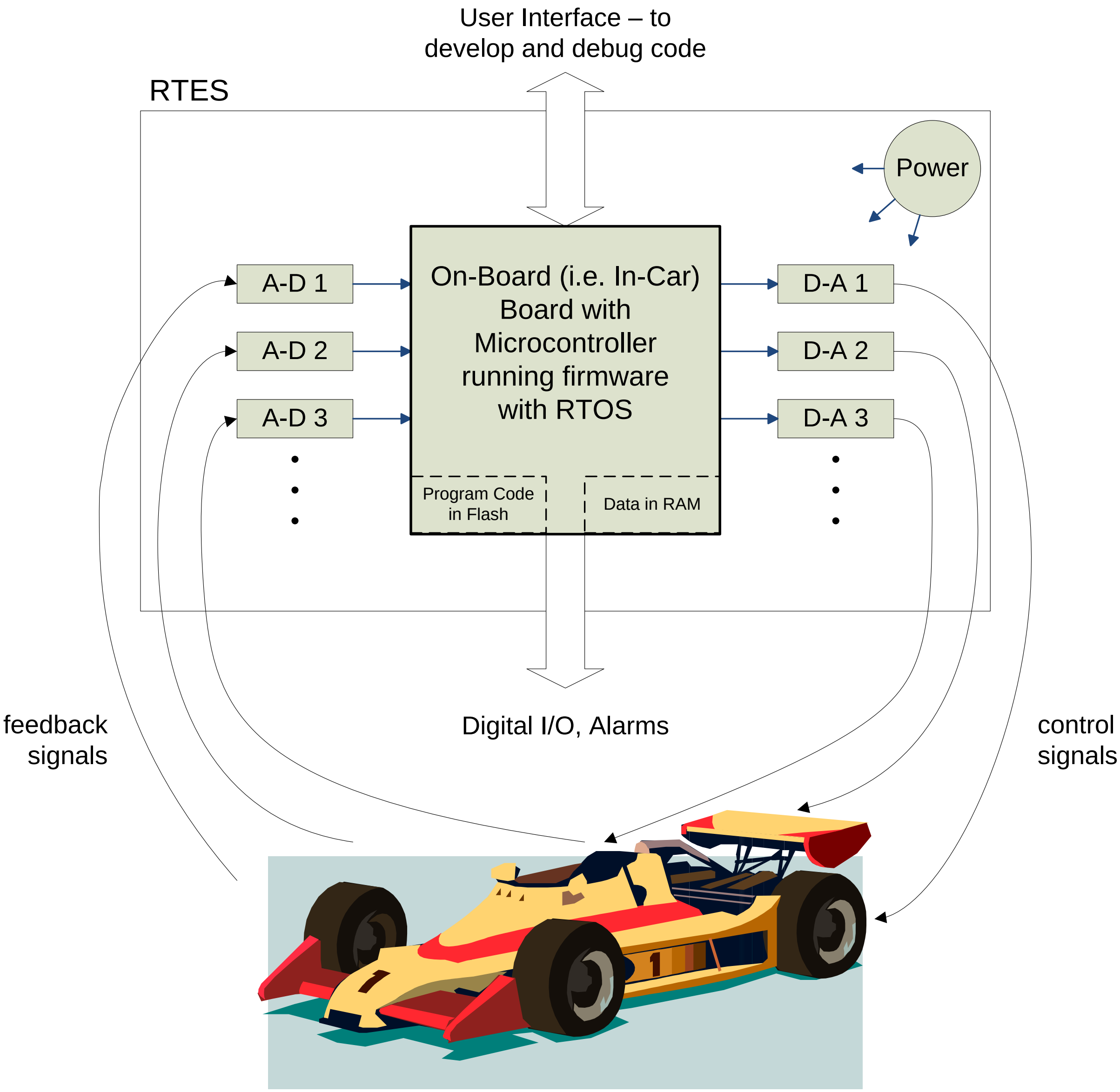
- a system unto itself; handles most everything for the task at hand
- system dedicated to application, usually not a platform for diverse/many applications
- autonomous, not dependently linked to some other system

Define "***Real-Time***"

- respond to external events in a timely fashion; keep up with external events

Example of Real-Time Embedded System

Race-Car Control System (Simplified Diagram)



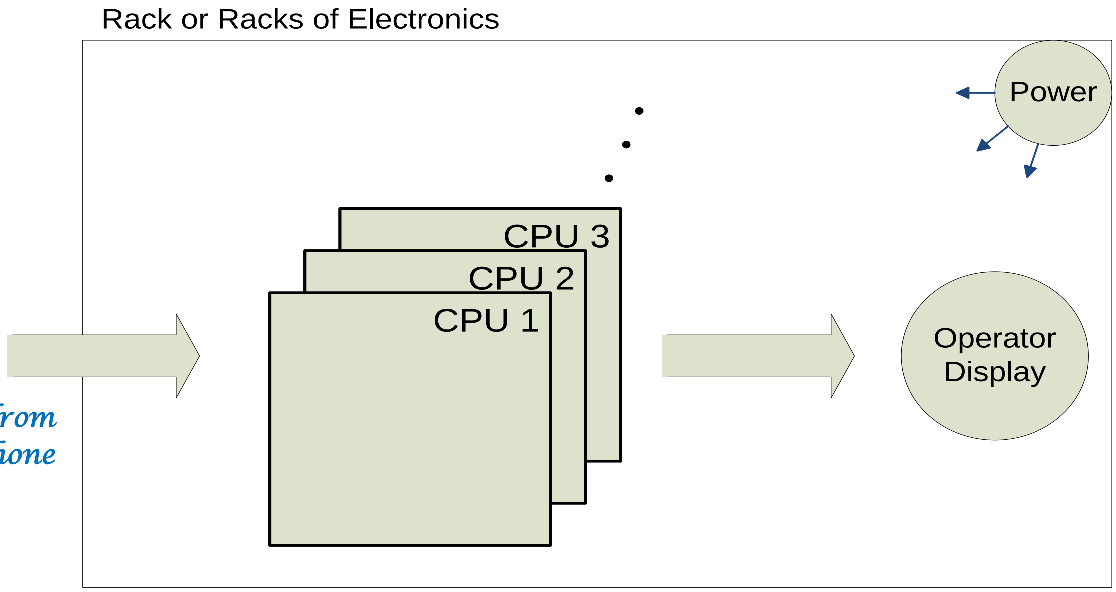
Program runs for entire time car is on:

- monitors engine emissions and adjusts fuel mixture in real-time
- monitors steering and adjusts handling in real-time
- monitors skidding and adjusts braking in real-time

Example of Real-Time Embedded System

Sonar Signal Processing System (Simplified Diagram)

underwater "towed-array"
- *approx 100 hydrophones each with A-D converter*

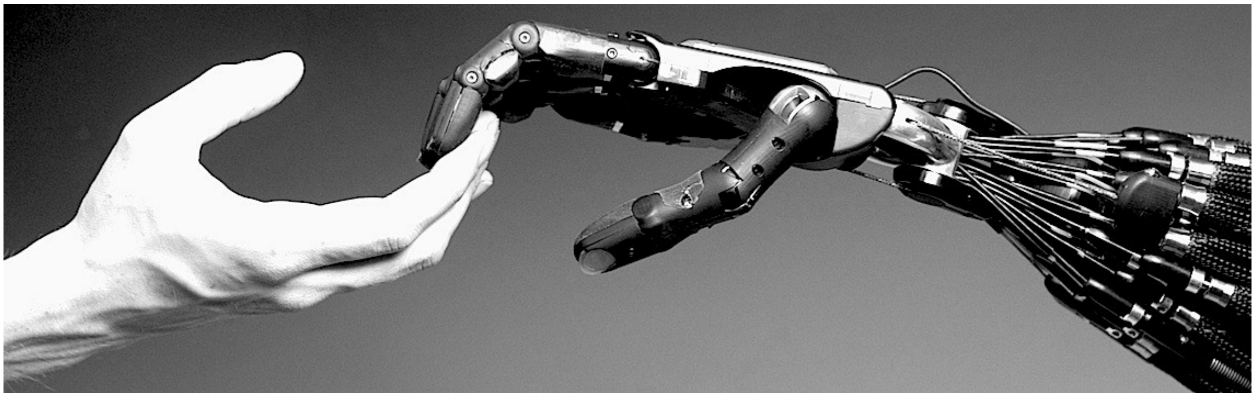


data from each hydrophone

Processors run 24/7 using DSP techniques to:
- apply 2-D FFTs to data matrix to form spatial "beams"
- post-process beam data to find any "hits"

Example of Real-Time Embedded System

Robotic Hand



The Shadow Robot Company

One hand w/forearm has:

- 9 A-D converters
- 25 joint position sensors
- 20 motors
- 20 temperature sensors
- 20 current sensors
- 40 Hall sensors

Example of Real-Time Embedded System

Skytrain Braking System

Test train with Coquitlam's Lincoln station in the background
022

The system must brake normally to stop at stations and also brake suddenly when an object is on the track.

What considerations went into the design of the braking system?

What firmware and hardware design approaches went into making the system highly reliable?

What are some other examples of Real-Time Embedded Systems

in the world around us?

A good RTES designer knows:

- hardware
  - processor chip (or sometimes multiple processor chips)
  - interaction with other hardware
  - board layout
- firmware
  - assembly instruction set
  - higher-level language, e.g. C
- other
  - cost issues
  - environmental issues
  - enclosures
  - algorithms – how to devise them, how to implement them

More Detailed Definition of Real-Time Embedded System:

Define “*Embedded System*”

- a system unto itself; handles most everything for the task at hand
- system dedicated to application, not a platform for diverse/many applications
- hardware is almost always based on a digital core
- often has software component referred to as “firmware”
- autonomous, not dependently linked to some other system
- often product runs on its own
- can be hidden from the user
- platform could be one of or a combination of:
    - discrete hardware
    - ASIC
    - FPGA
    - DSP
    - µP
- if based on a processor(s), software can be C, assembly, C++, or combination thereof; single OS

Define “*Real-Time*”

- respond to external events in a timely fashion; keep up with external events
- “correct and timely responses”
    - low latency
    - steady queue
    - in the “real” world
- often requires high-speed h/w and s/w capable of fast response, fast action
- can have “hard” vs “soft” timing constraints

Hard vs Soft Timing Constraints

Hard

e.g. skytrain automatic braking system
when object on track

start
task
end
task

Soft

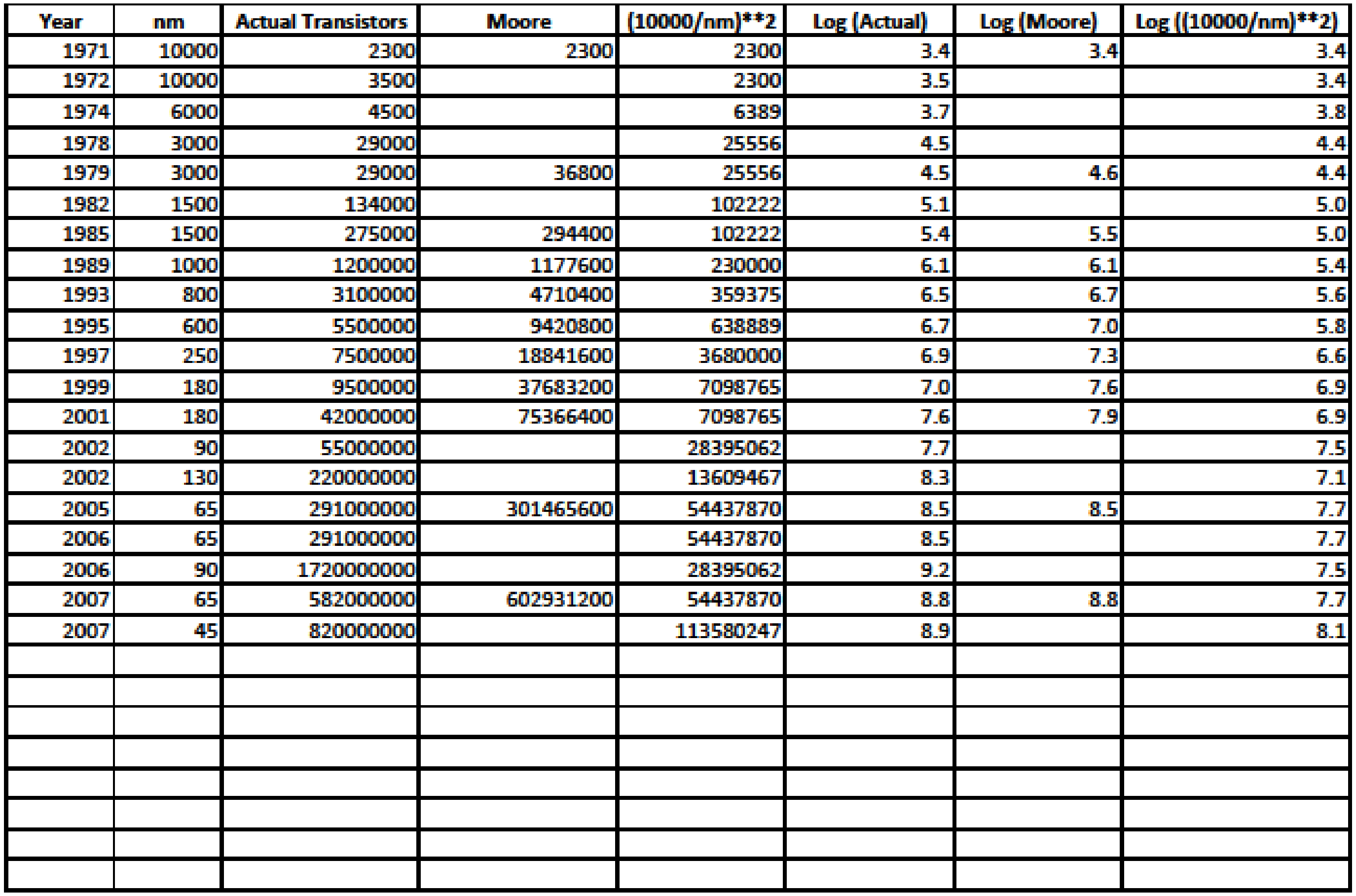
e.g. telephone switch

start
task
end
task

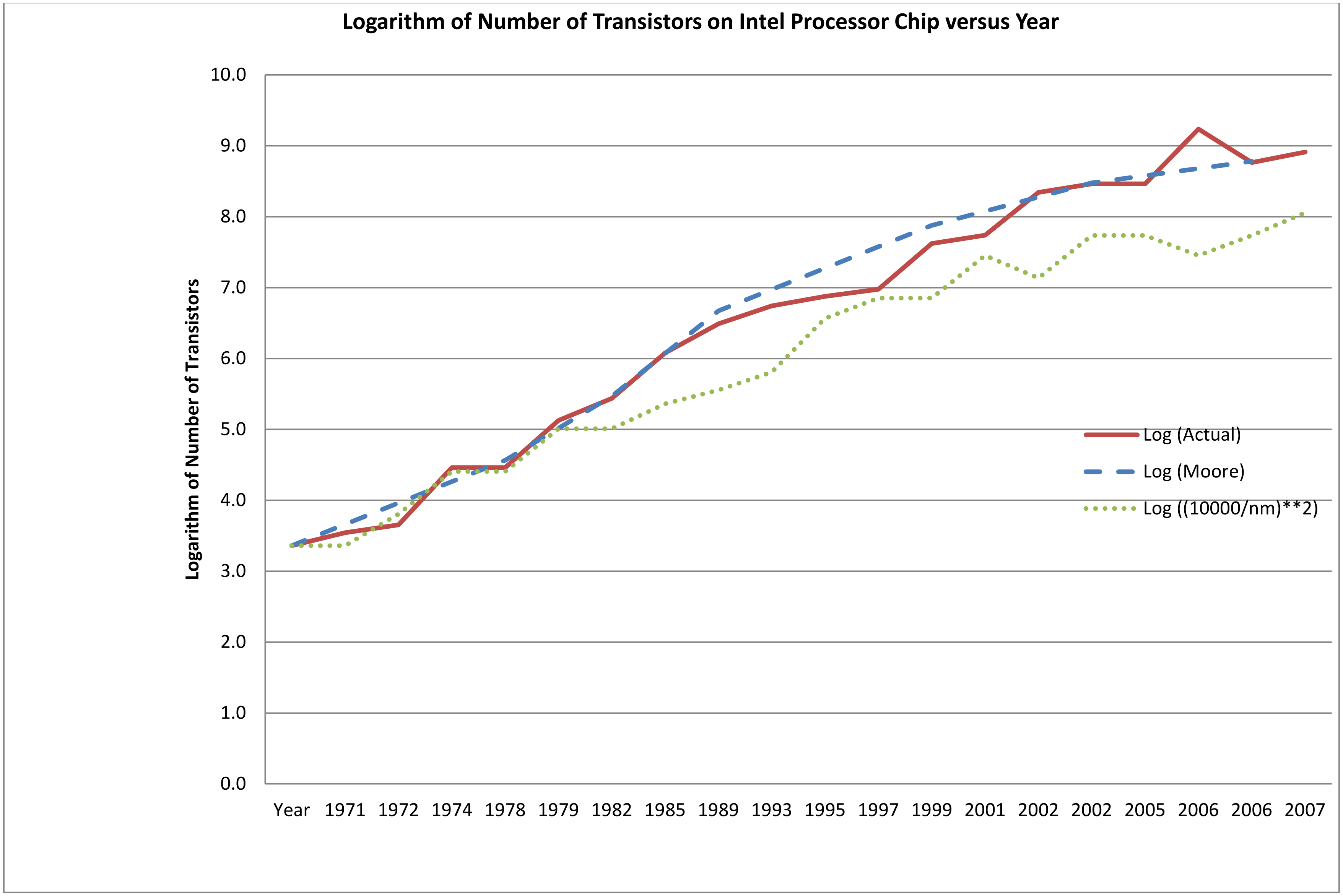
Chronological History of Intel Microprocessor

Date	Device	Comment	Clock Rate[1]	Data Bus Size	Address Bus Size	#Transistors	Junction Width	Chip Package
1971	4004	4-bit processor (desktop calculator mfr'd by Japanese company)	108 kHz	4	12	2300	10µ	16-pin DIP
1972	8008		500 kHz	8	14	3500	10µ	18-pin DIP
1974	8080		2 MHz	8	16	4500	6µ	40-pin DIP
1979	8088	**basis of first PC** (mfr'd by IBM)	5 MHz	16 internal 8 external	20	29000	3µ	40-pin DIP
1982	80286		6 MHz	16	24	134000	1.5µ	68-PGA
1985	80386		16 MHz	32	32	275000	1.5µ	132-PGA
1989	80486	introduced on-chip **floating-point unit**	25 MHz	32	32	1200000	1µ	169-PGA
1993	Pentium	"Pent" name derived from "5th" generation	66 MHz	64	32	3100000	0.8µ	273-PGA
2005	Pentium D	introduced **two execution cores** on one chip	3.2 GHz	64	32	291000000	65nm	775-LGA
2007	Core 2 Quad	introduced four execution cores on one chip	2.66 GHz	64	32	582000000	65nm	775-LGA

[1] This is the initial clock rate when the chip first was introduced. It generally increased as newer versions were created. Ditto for the other tables herein.

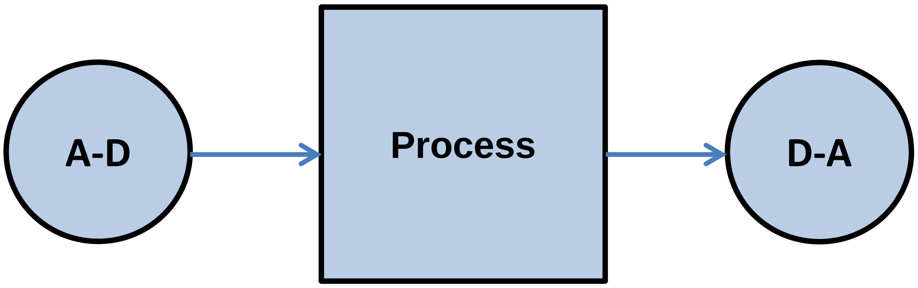


Year	nm	Actual Transistors	Moore	(10000/nm)**2	Log (Actual)	Log (Moore)	Log ((10000/nm)**2)
1971	10000	2300	2300	2300	3.4	3.4	3.4
1972	10000	3500		2300	3.5		3.4
1974	6000	4500		6389	3.7		3.8
1978	3000	29000		25556	4.5		4.4
1979	3000	29000	36800	25556	4.5	4.6	4.4
1982	1500	134000		102222	5.1		5.0
1985	1500	275000	294400	102222	5.4	5.5	5.0
1989	1000	1200000	1177600	230000	6.1	6.1	5.4
1993	800	3100000	4710400	359375	6.5	6.7	5.6
1995	600	5500000	9420800	638889	6.7	7.0	5.8
1997	250	7500000	18841600	3680000	6.9	7.3	6.6
1999	180	9500000	37683200	7098765	7.0	7.6	6.9
2001	180	42000000	75366400	7098765	7.6	7.9	6.9
2002	90	55000000		28395062	7.7		7.5
2002	130	220000000		13609467	8.3		7.1
2005	65	291000000	301465600	54437870	8.5	8.5	7.7
2006	65	291000000		54437870	8.5		7.7
2006	90	1720000000		28395062	9.2		7.5
2007	65	582000000	602931200	54437870	8.8	8.8	7.7
2007	45	820000000		113580247	8.9		8.1


Logarithm of Number of Transistors on Intel Processor Chip versus Year
Logarithm of Number of Transistors
10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0.0
Year
1971
1972
1974
1978
1979
1982
1985
1989
1993
1995
1997
1999
2001
2002
2002
2005
2006
2006
2007
Log (Actual)
Log (Moore)
Log ((10000/nm)**2)

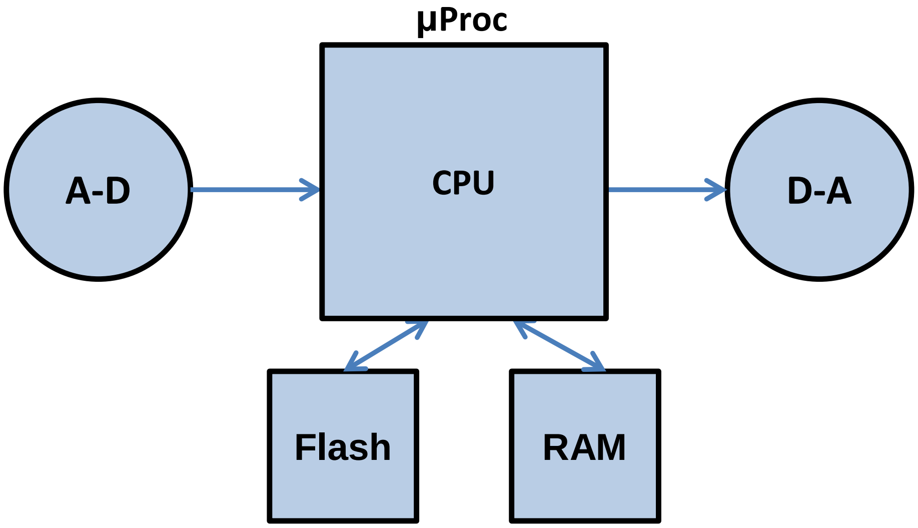
Hardware Platforms for the Ubiquitous Embedded DSP System

Consider the ubiquitous embedded DSP system:



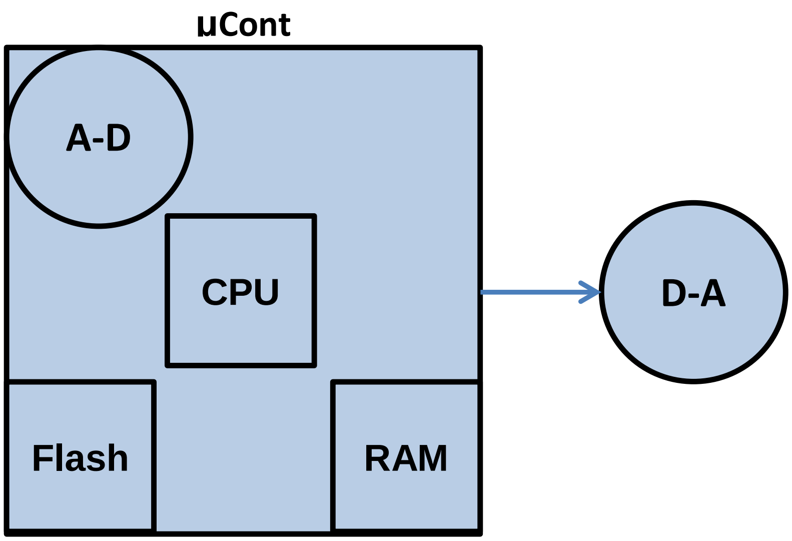
The following are possible hardware platforms to implement the above system…

µProc (Microprocessor)



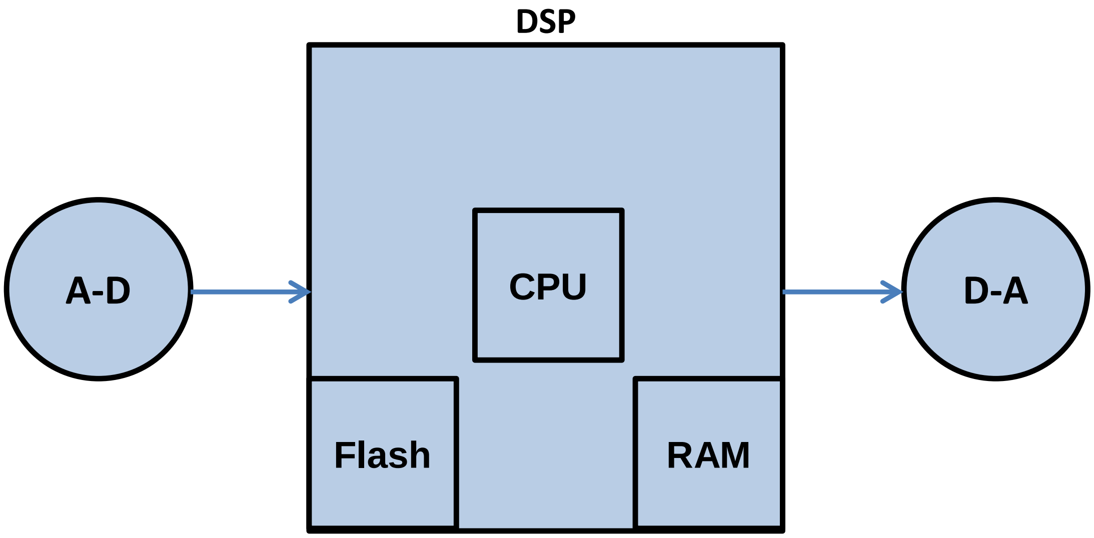
advantages *disadvantages*

µCont (Microcontroller)



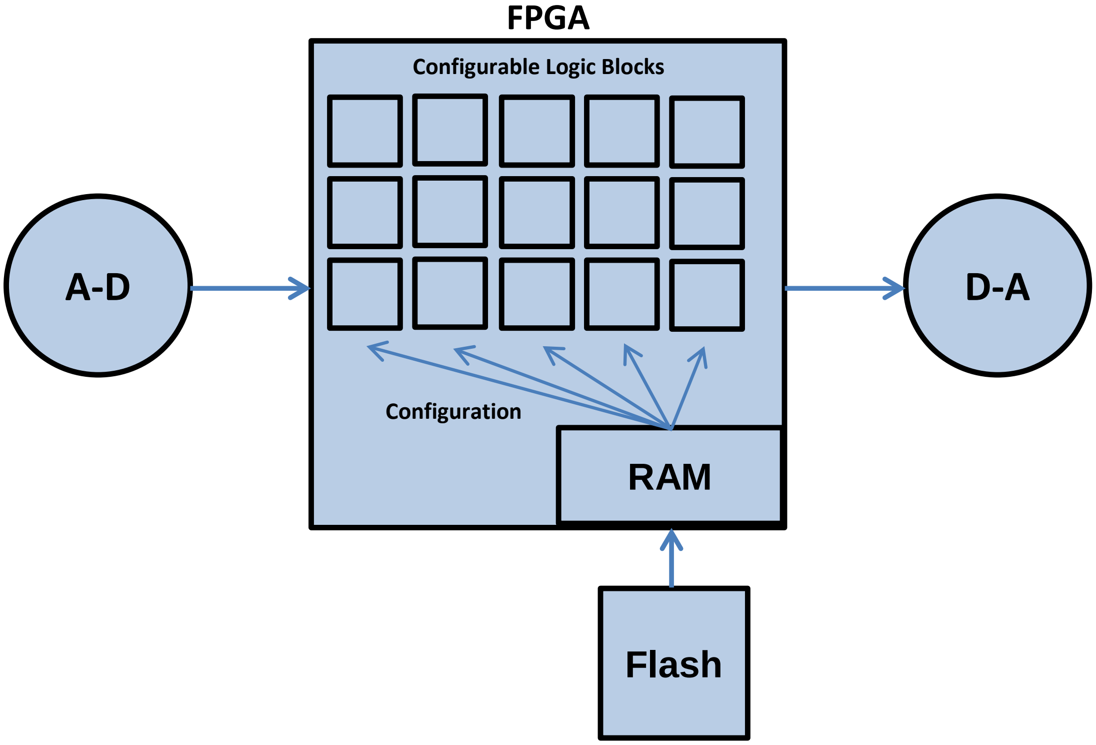
advantages *disadvantages*

DSP (Digital Signal Processor)



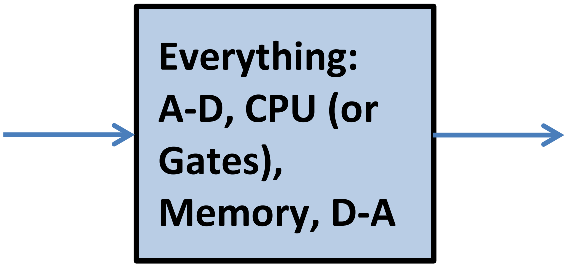
advantages *disadvantages*

FPGA (Field-Programmable Gate Array)



advantages *disadvantages*

Application-Specific Integrated Circuit (ASIC)



advantages *disadvantages*

Selected Chronological History of Digital Signal Processor (DSP) Chip

Date	Device	Creator	Comment	Parallel Multiplier	Clock Rate	Instruction Cycle Time	Program Memory Size	Data Memory Size	Chip Package
1978	S2811	AMI	not successful	12x12 fixed-point	20 MHz	300 nsec	256 ROM	128 RAM	
1980	µPD7720	NEC (now Renesas)	**first commercial DSP**	16x16 fixed-point	8 MHz	250 nsec	512 ROM	128 RAM	28-pin DIP
1982	TMS32010	TI	introduced modified **Harvard** architecture	16x16 fixed-point	20 MHz	200 nsec	1536 ROM or 4K external	144 RAM	40-pin DIP
1984	HD61810	Hitachi	introduced on-chip **floating-point capability**	16x16 fixed-point	16 MHz	250 nsec	512 ROM	200 RAM	40-pin DIP
1985	TMS32020	TI	introduced **multiple-accumulate** instruction	16x16 fixed-point	20 MHz	200 nsec	256 RAM	256 RAM	68-PGA
1980s	µPD77230	NEC (now Renesas)		32x32 floating-point	13.3 MHz	150 nsec	2K ROM or 4K external	1024 RAM	68-PGA
1990s	56000	Motorola (now Freescale)	introduced **dual data memory banks**	24x24 fixed-point	33 MHz	60.6 nsec	512 RAM (boot from external)	256 X RAM 256 Y RAM	88-PGA
1990s	TMS320C30	TI		32x32 floating-point	33 MHz	74 nsec	16M external	1024 X RAM 1024 Y RAM	181-PGA
2011	TMS320C6671	TI	introduced **VLIW** instructions with **8 ALUs**	32x32 fixed-point (each)	1 GHz	1 nsec	approx 512K+ shared		841-BGA
2011	TMS320VC5510	TI	**very low power**	17x17 fixed-point	50 MHz	20 nsec	320K program/data		144-BGA

Selected Chronological History of Microcontroller (µC) Chip

Date	Device	Creator	Comment	Parallel Multiplier	Clock Rate	Instruction Cycle Time	Program Memory Size	Data Memory Size	Chip Package
1974	TMS1000	TI	4-bit CPU	no	300 kHz	?	1K ROM	32 RAM	28-pin DIP
1980	8051	Intel	8-bit CPU	no	12 MHz	1 µsec	4K EPROM (8748) 4K PROM (8048)	128 RAM	40-pin DIP
1993	PIC16x84	Microchip	8-bit CPU introduced on-chip **EEPROM** program memory	no	10 MHz	100 nsec	1K Flash	68 RAM 64 EEPROM	18-pin DIP
late 1990s	68HC12	Motorola (now Freescale)	16-bit CPU	16x16 MAC takes 13 cycles	8 MHz	250 nsec	128K Flash	8K RAM	112-TQFP
2008	PIC32MX320	Microchip	32-bit CPU 5-stage pipeline	32x32 fixed-point (takes 2 cycles)	40 MHz	25 nsec	32K Flash	8K RAM	64-TQFP

Selected Chronological History of Microcontroller w/DSP (µC w/DSP) Chip

Date	Device	Creator	Comment	Parallel Multiplier	Clock Rate	Instruction Cycle Time	Program Memory Size	Data Memory Size	Chip Package
2001	dsPIC30F1010	Microchip	16-bit CPU	16x16 fixed-point **(in one cycle)**	120 MHz	33.3 nsec	6K Flash	256 RAM	28-pin DIP
2009	TMS320F28027 Piccolo	TI	32-bit CPU very low cost low power up to 16 A-D channels SPI,I^{2}C,SCI ports	32x32 fixed-point (in one cycle)	60 MHz (and up)	16.6 nsec	up to 64K Flash	up to 12K RAM	48-LQFP