EE319K Introduction to Embedded Systems

- Course description
- Introduce embedded microcomputer systems,
- Flowcharts,
- Data flow graphs,
- Call graphs.

Useful web sites

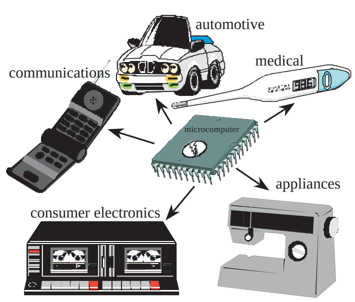
Valvano's web page	http://users.ece.utexas.edu/~valvano
EE319K material	http://users.ece.utexas.edu/~valvano/EE319K
Data sheets	http://users.ece.utexas.edu/~valvano/Datasheets/
Starter files	http://users.ece.utexas.edu/~valvano/Starterfiles/
Video lessons	http://users.ece.utexas.edu/~valvano/Lessons/
TExaS lessons	http://users.ece.utexas.edu/~valvano/Readme.htm
C programming	http://users.ece.utexas.edu/~ryerraballi/CPrimer/

Information about the 9S12 board

Physical layout	http://users.ece.utexas.edu/~valvano/Datasheets/TechArts9S12DP512.pdf
Circuit diagram	http://users.ece.utexas.edu/~valvano/Datasheets/AD9S12DP512R1.pdf
9S12DG128 data sheet	http://users.ece.utexas.edu/~valvano/Datasheets/MC9S12DG128.pdf
Large 9S12 manual	http://users.ece.utexas.edu/~valvano/Datasheets/S12CPUV2.pdf
Small 9S12 manual	http://users.ece.utexas.edu/~valvano/Datasheets/CPU12rg.pdf

An *embedded computer system* includes a microcomputer
mechanical, chemical and electrical devices
specific dedicated purpose, and
packaged up as a complete system.

- communications,
- automotive,
- military,
- medical,
- consumer,
- machine control.



Each **embedded microcomputer** system
accepts inputs,
performs calculations, and
generates outputs
runs in "real time."

In a **real time system**, upper bound on the time required to
perform the input/calculation/output
respond to external events

Because of the real time nature of these systems, we will study the rich set of features built into these microcontrollers to handle all aspects of time.

1.2. Attitude
clients
coworkers

Test it now.
Plan for testing.
Get help.

There is just no simple way to get to the moon.

Use our creativity to break a complex problem into simple components, rather than developing complex solutions to simple problems.

1.3. Components of an embedded system

embedded "hidden inside so one can't see it."
computer
processor

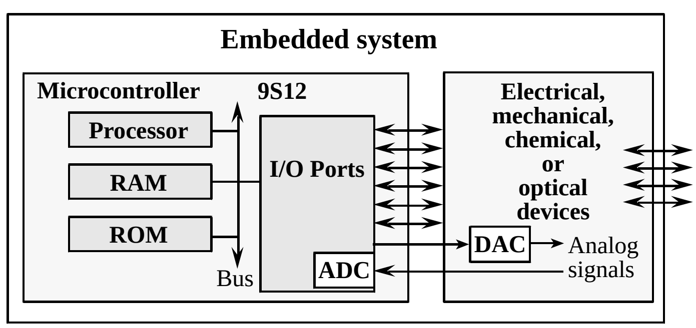


Figure 1.1. An embedded system includes a microcontroller interfaced to external devices.

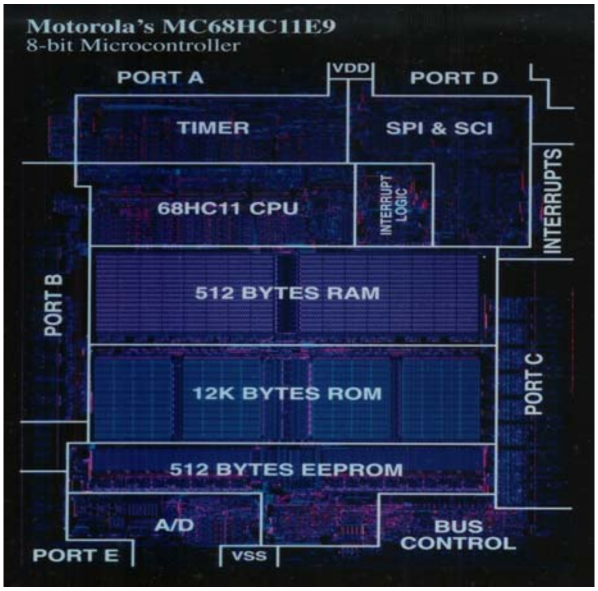
Address	Size	Device	Device	Contents
$0000 to $03FF	1 KiB	I/O ports	Input/output devices	Access external devices
$0400 to $07FF	1 KiB	EEPROM	Electrically erasable PROM	Fixed constants
$2000 to $3FFF	8 KiB	RAM	Random Access Memory	Variables and stack
$4000 to $FFFF	48 KiB	EEPROM	Electrically erasable PROM	Programs and fixed constants

The 9S12DG128 has 128 Kibibytes of EEPROM and 8 Kibibytes of RAM.

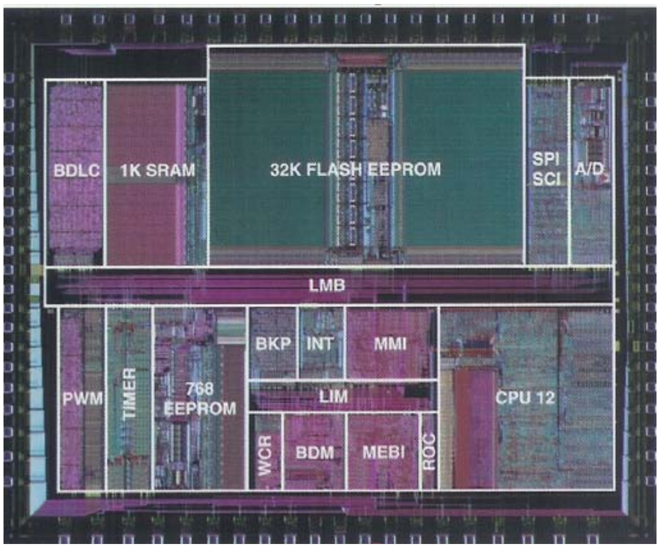
Address	Size	Device	Device	Contents
$0000 to $03FF	1 KiB	I/O ports	Input/output devices	Access external devices
$0400 to $07FF	1 KiB	EEPROM	Electrically erasable PROM	Fixed constants
$0800 to $3FFF	14 KiB	RAM	Random Access Memory	Variables and stack
$4000 to $FFFF	48 KiB	EEPROM	Electrically erasable PROM	Programs and fixed constants

Table 2.3. The 9S12DP512 has 512 Kibibytes of EEPROM and 14 Kibibytes of RAM.

microcomputer
microcontroller



Courtesy: Prof. Gary Daniels



68HC12 Die Photo

interface
- parallel - data is available simultaneously on groups of lines
- serial - binary data is available one bit at a time on a single line
- analog - data is encoded as a variable voltage
- time - data is encoded as period, frequency, pulse width or phase shift

nonvolatile
port
device driver

1.4. Flowcharts

Design a flowchart for a cruise controller.
Goal: Drive at constant speed
- Inputs: Speed, DesiredSpeed, *SpeedLimit, …*
- Outputs: GasPedal, *brake, …*

How will I test it?
- Measure speed compare to desired speed

How do I start programming?
- What tasks are done once at beginning?
- What tasks do I do over and over in the middle?
- What tasks are done once at end?

1) Draw a data flow graph
2) Define modules and connect together (call graph)
- Current speed input module
- Desired speed input module
- GasPedal output module
- Controller
- *Front panel display*
- *Safety black box*

3) Software design
- Flowcharts
- Data formats
- Pseudocode

4) Software implementation four approaches
- Inside out
- Time sequential
- Top-down
- Bottom-up

structured programming.
- **sequence**,
- **conditional**
- **while-loop**
- **fork-join**
- **interrupt**

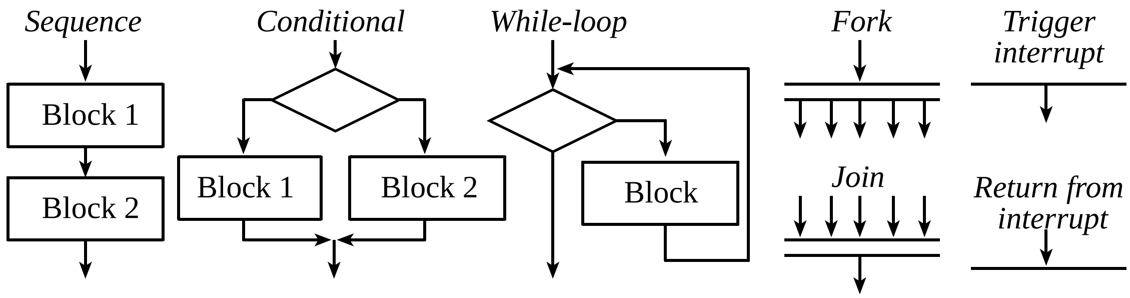
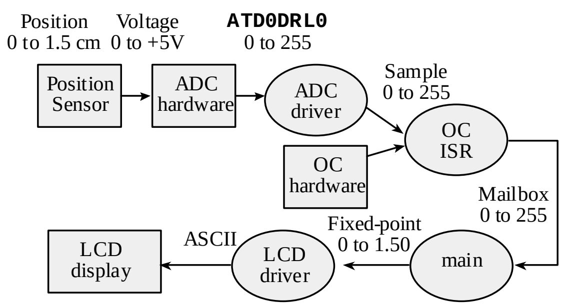
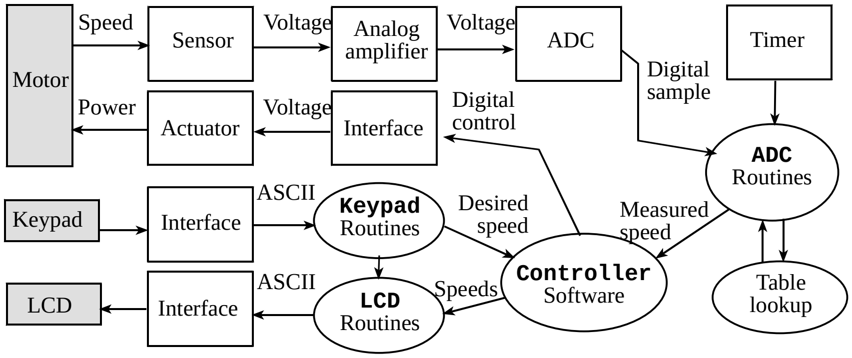


Figure 1.3. Flowchart showing the basic building blocks of structured programming.

data flow graph (dependency graph)



Lab 6. A data flow graph showing how the position signal passes through the system.



A data flow graph showing how signals pass through a motor controller.

Call-graphs

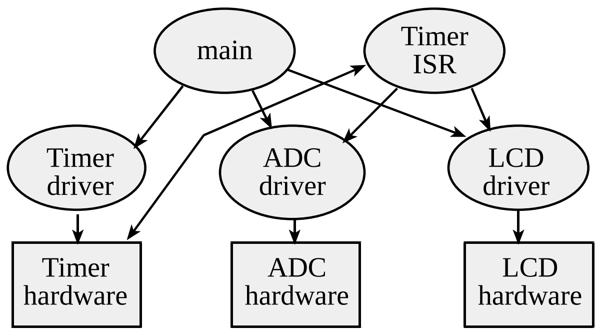


Figure 1.7. A call flow graph for a simple position measurement system.

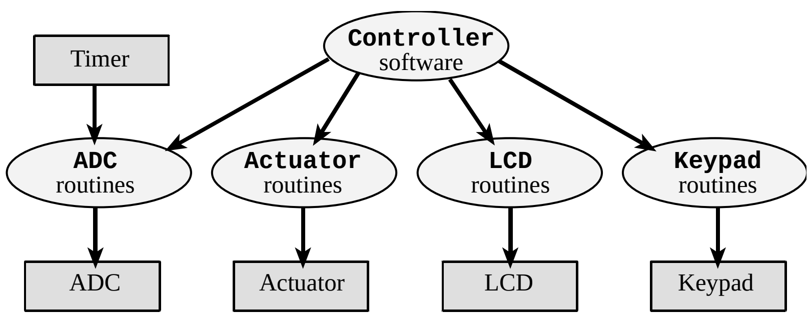


Figure 1.10. A call flow graph for a motor controller.

Testing

validate basic functions
static efficiency (memory requirements),
dynamic efficiency (execution speed),
accuracy (difference between truth and measured), and
stability (consistent operation.)

Maintenance

correcting mistakes,
adding new features,
optimizing for execution speed or program size,
porting to new computers or operating systems, and
reconfiguring the system to solve a similar problem.

Golden Rule of Software Development
Write software for others as you wish they would write for you.

1.6.2. Qualitative Performance Measurements

Can we prove our software works?
Is our software easy to understand?
Is our software easy to change?

self-documented code,
abstraction,
modularity, and
layered software.

You can tell if you are a good programmer if
1) you can understand your own code 12 months later, 2) others can make changes to your code.

The bottom line
Definitions terms in your technical communication
RAM is expensive in an embedded system
Data flow graph is used during design phase