Introduction to Embedded Systems Design

Ravi Suppiah
Lecturer, NUS SoC

The Architecture for the Digital World® ARM

Introduction

- What is an Embedded System?
  - Application-specific computer system
  - Built into a larger system
- Why add a computer to the larger system?
  - Better performance
  - More functions and features
  - Lower cost
  - More dependability
- Economics
  - Microcontrollers (used for embedded computers) are high-volume, so recurring cost is low
  - "Nonrecurring" cost dominated by software development
- Networks
  - Often embedded system will use multiple processors communicating across a network to lower parts and assembly costs and improve reliability

Options for Building Embedded Systems

Dedicated Hardware

Software Running on Generic Hardware

Implementation	Design Cost	Unit Cost	Upgrades & Bug Fixes	Size	Weight	Power	System Speed
Discrete Logic	low	mid	hard	large	high	?	very fast
ASIC	high ($500K/ mask set)	very low	hard	tiny - 1 die	very low	low	extremely fast
Programmable logic – FPGA, PLD	low	mid	easy	small	low	medium to high	very fast
Microprocessor + memory + peripherals	low to mid	mid	easy	small to med.	low to moderate	medium	moderate
Microcontroller (int. memory & peripherals)	low	mid to low	easy	small	low	medium	slow to moderate
Embedded PC	low	high	easy	medium	moderate to high	medium to high	fast

Example Embedded System: Bike Performance Tracker

- Functions
  - Speed and distance measurement
- Constraints
  - Size
  - Cost
  - Power and Energy
  - Weight
- Inputs
  - Wheel rotation indicator
  - Mode key
- Output
  - Liquid Crystal Display
- Low performance MCU
  - 8-bit

Gasoline Automobile Engine Control Unit

- Functions
  - Fuel injection
  - Air intake setting
  - Spark timing
  - Exhaust gas circulation
  - Electronic throttle control
  - Knock control
- Constraints
  - Reliability in harsh environment
  - Cost
  - Weight

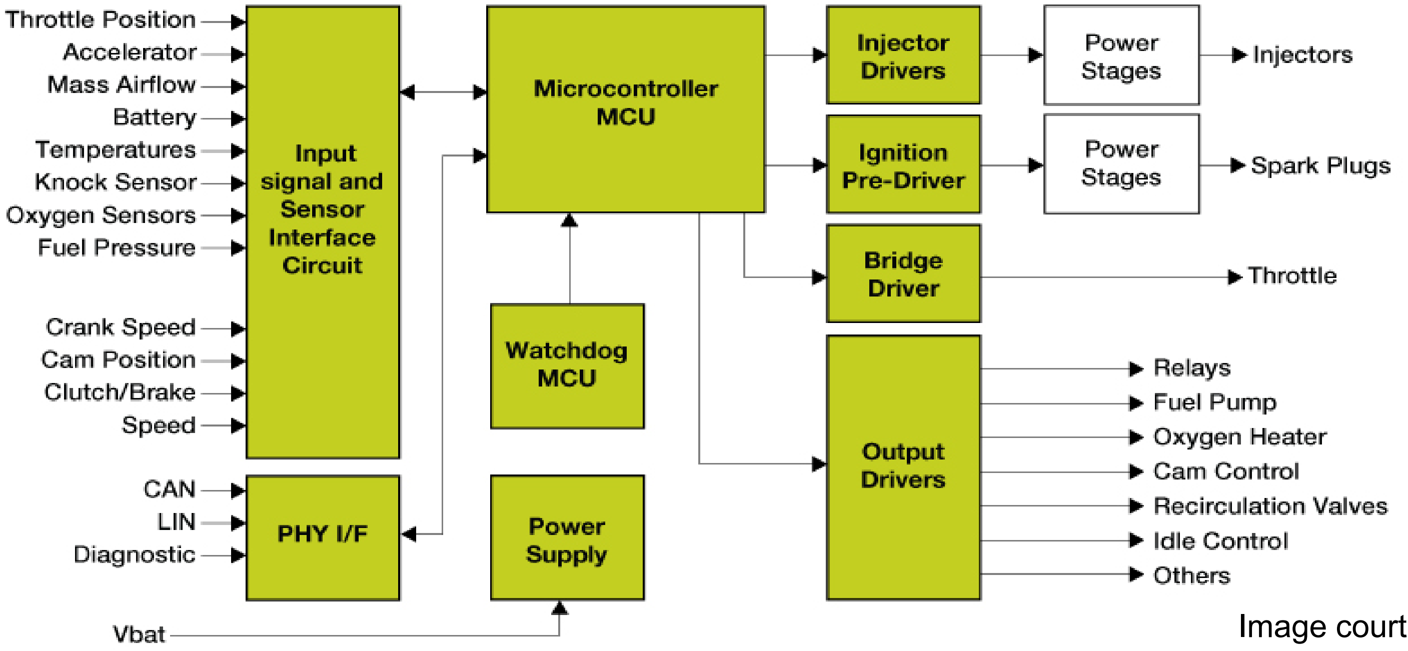


Image courtesy of Freescale

- Many Inputs and Outputs
  - Discrete sensors & actuators
  - Network interface to rest of car
- High Performance MCU
  - 32-bit, 3 MB flash memory, 150 - 300 MHz

Benefits of Embedded Computer Systems

- Greater performance and efficiency
  - Software makes it possible to provide **sophisticated control**

- Lower costs
  - Less expensive components can be used
  - Manufacturing costs reduced
  - Operating costs reduced
  - Maintenance costs reduced

- More features
  - Many not possible or practical with other approaches

- Better dependability
  - Adaptive system which can compensate for failures
  - Better diagnostics to improve repair time

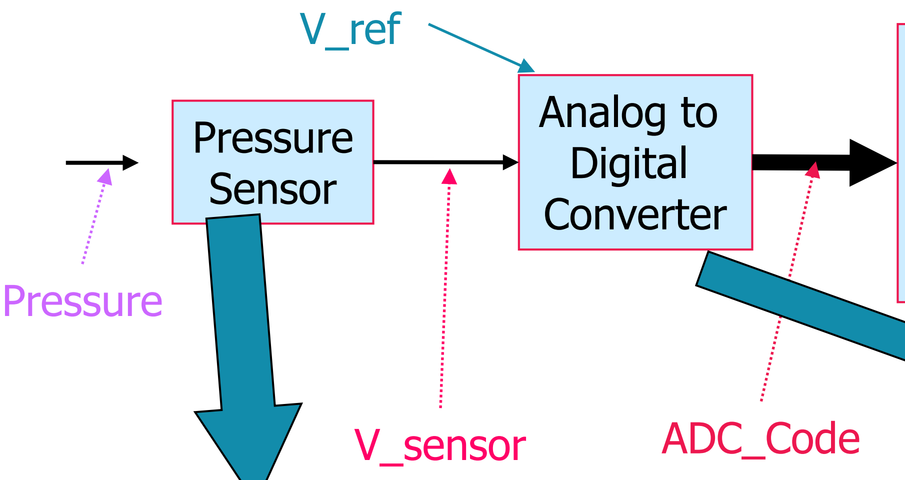
Embedded System Functions

- Closed-loop control system
  - Monitor a process, adjust an output to maintain desired set point (temperature, speed, direction, etc.)
- Sequencing
  - Step through different stages based on environment and system
- Signal processing
  - Remove noise, select desired signal features
- Communications and networking
  - Exchange information reliably and quickly

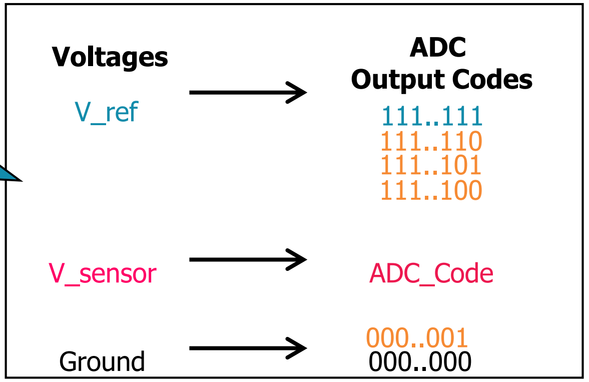
Attributes of Embedded Systems

- Interfacing with larger system and environment
  - Analog signals for reading sensors
    - Typically use a voltage to represent a physical value
  - Power electronics for driving motors, solenoids
  - Digital interfaces for communicating with other digital devices
    - Simple - switches
    - Complex - displays

Example Analog Sensor - Depth Gauge



```
// Your software
ADC_Code = ADC0->R[0];
V_sensor = ADC_code*V_ref/1023;
Pressure_kPa = 250 * (V_sensor/V_supply+0.04);
Depth_ft = 33 * (Pressure_kPa – Atmos_Press_kPa)/101.3;
```



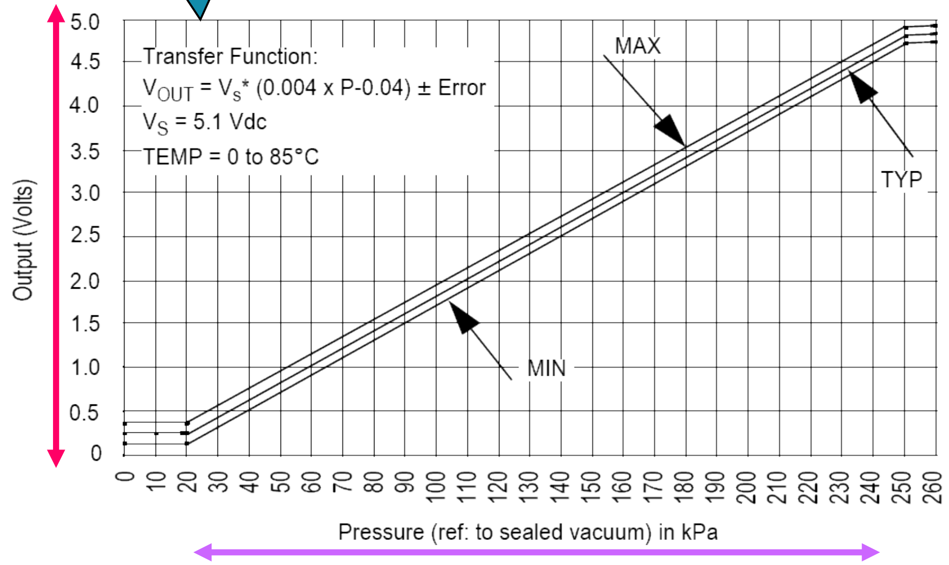
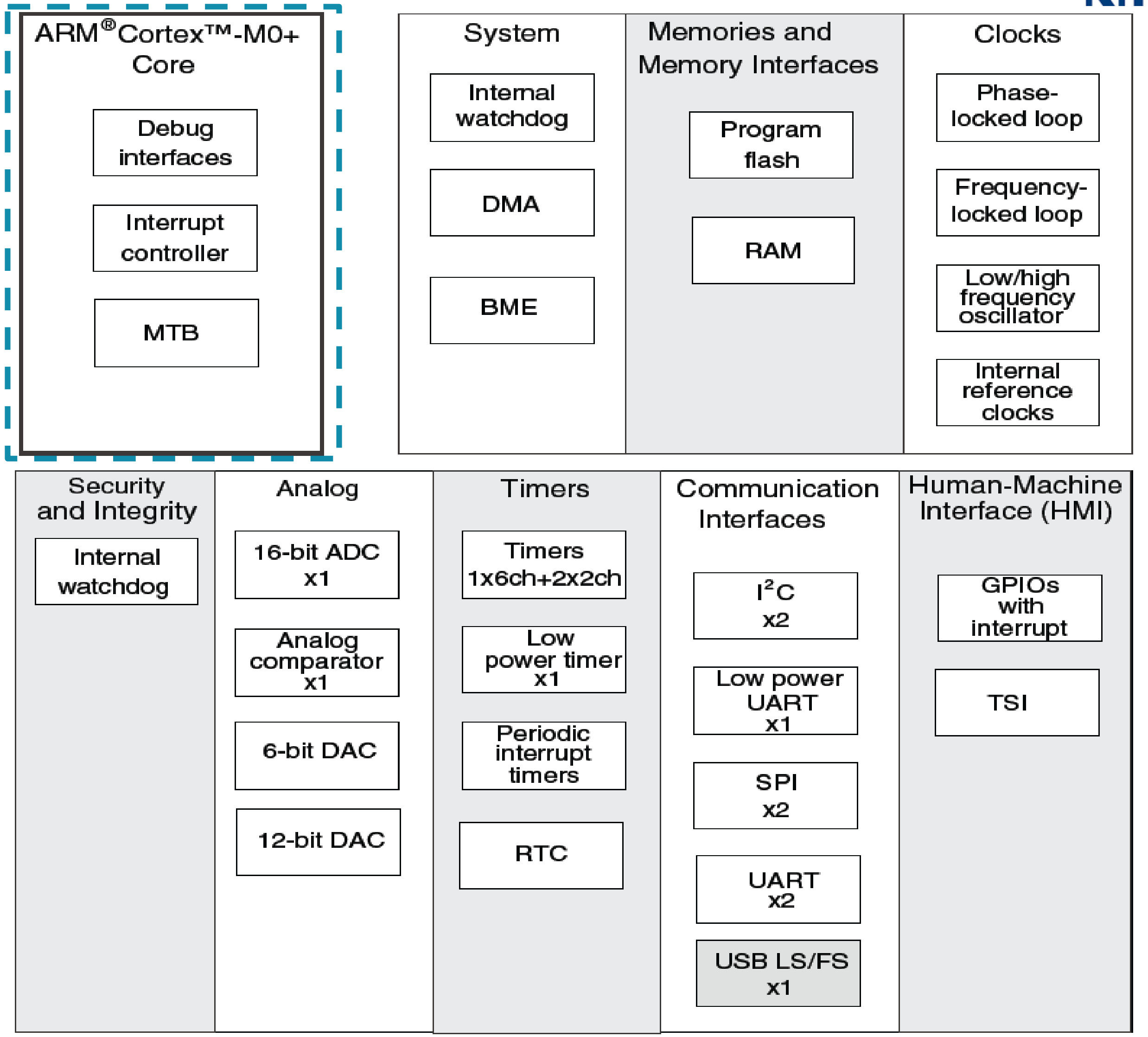


Figure 4. Output vs. Absolute Pressure

1. Sensor detects *pressure* and generates a proportional *output voltage* V_sensor
2. ADC generates a proportional digital *integer* (code) based on V_sensor and V_ref
3. Code can convert that integer to a something more useful
    1. first a float representing the *voltage*,
    2. then another float representing *pressure*,
    3. finally another float representing *depth*

Microcontroller vs. Microprocessor

- Both have a CPU core to execute instructions
- Microcontroller has peripherals for concurrent embedded interfacing and control
  - Analog
  - Non-logic level signals
  - Timing
  - Clock generators
  - Communications
    - point to point
    - network
  - Reliability and safety



ARM®Cortex™-M0+ Core

Debug interfaces

Interrupt controller

MTB

System

Internal watchdog

DMA

BME

Memories and Memory Interfaces

Program flash

RAM

Clocks

Phase-locked loop

Frequency-locked loop

Low/high frequency oscillator

Internal reference clocks

Security and Integrity

Internal watchdog

Analog

16-bit ADC x1

Analog comparator x1

6-bit DAC

12-bit DAC

Timers

Timers 1x6ch+2x2ch

Low power timer x1

Periodic interrupt timers

RTC

Communication Interfaces

I²C x2

Low power UART x1

SPI x2

UART x2

USB LS/FS x1

Human-Machine Interface (HMI)

GPIOs with interrupt

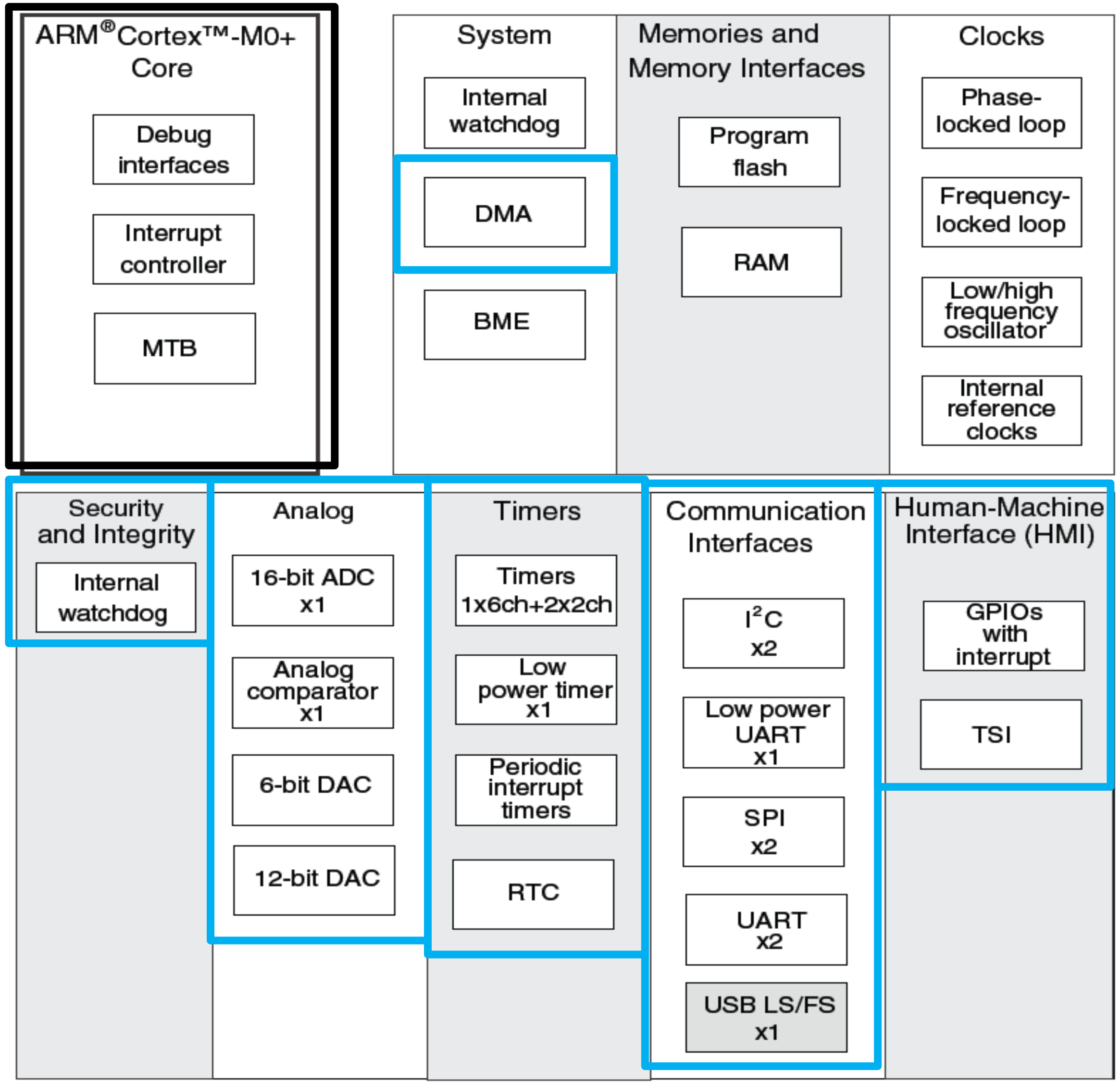
TSI

Attributes of Embedded Systems

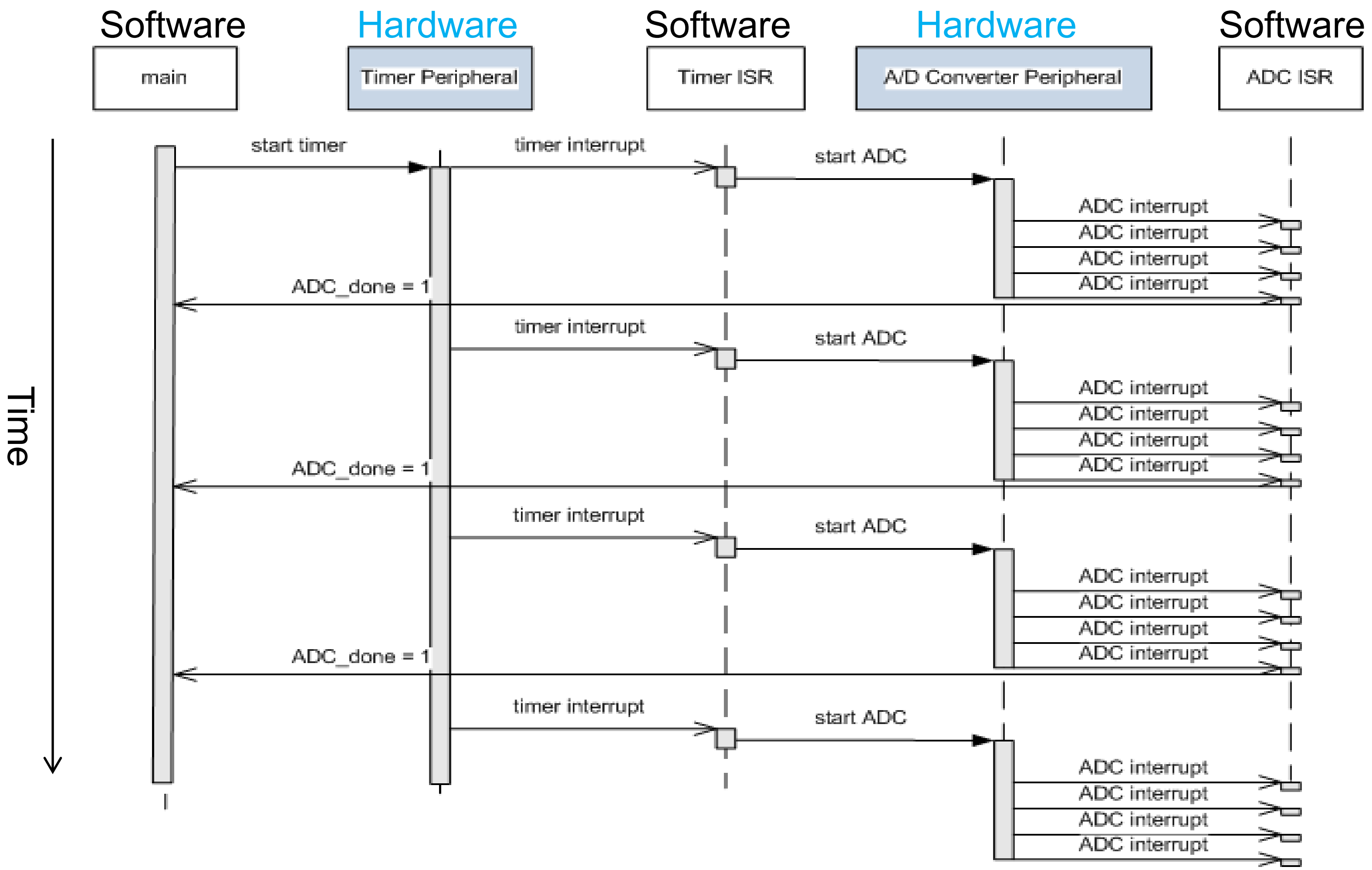
- Concurrent, reactive behaviors
  - Must respond to sequences and combinations of events
  - Real-time systems have deadlines on responses
  - Typically must perform multiple separate activities concurrently

MCU Hardware & Software for Concurrency

- CPU executes instructions from one or more thread of execution
- Specialized hardware peripherals add dedicated concurrent processing
  - DMA - transferring data between memory and peripherals
  - Watchdog timer
  - Analog interfacing
  - Timers
  - Communications with other devices
  - Detecting external signal events
- Peripherals use ***interrupts*** to notify CPU of events



Concurrent Hardware & Software Operation



- Embedded systems rely on both MCU ***hardware peripherals*** and ***software*** to get everything done on time

Attributes of Embedded Systems

- Fault handling
  - Many systems must operate independently for long periods of time, requiring system to handle likely faults without crashing
  - Often fault-handling code is larger and more complex than the normal-case code

- Diagnostics
  - Help service personnel determine problem quickly

Constraints

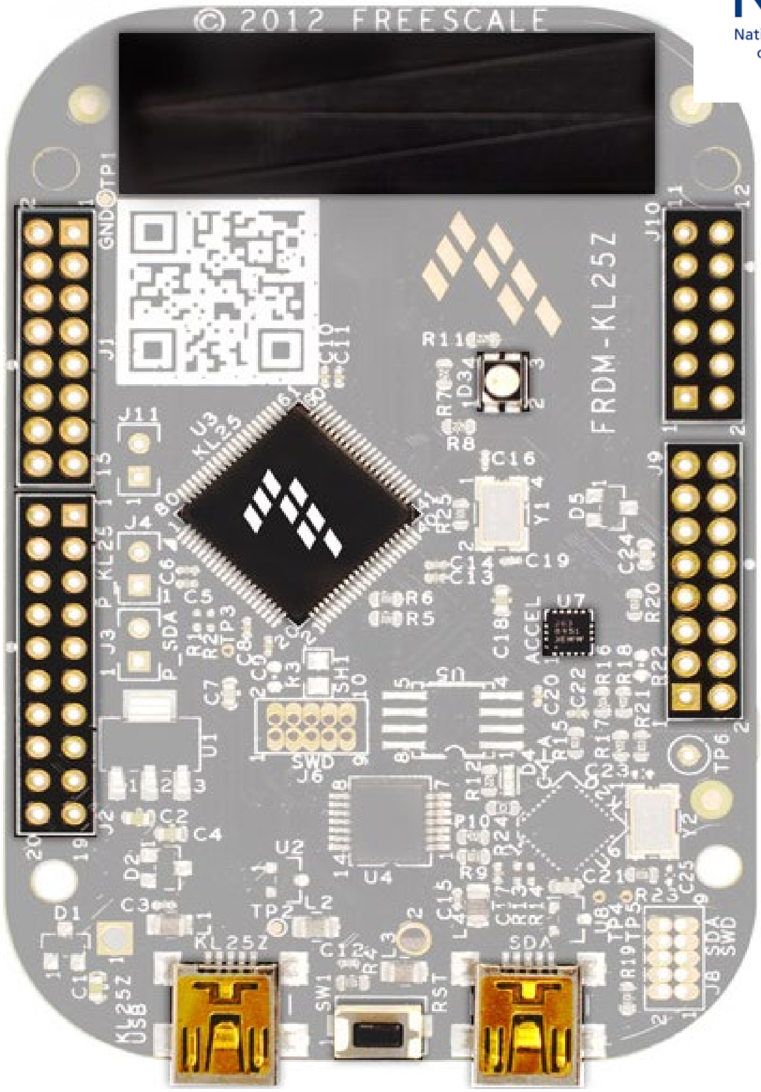
- Cost
  - Competitive markets penalize products which don’t deliver adequate value for the cost
- Size and weight limits
  - Mobile (aviation, automotive) and portable (e.g. handheld) systems
- Power and energy limits
  - Battery capacity
  - Cooling limits
- Environment
  - Temperatures may range from -40°C to 125°C, or even more

Impact of Constraints

- Microcontrollers used (rather than microprocessors)
  - Include peripherals to interface with other devices, respond efficiently
  - On-chip RAM, ROM reduce circuit board complexity and cost
- Programming language
  - Programmed in C rather than Java (smaller and faster code, so less expensive MCU)
  - Some performance-critical code may be in assembly language
- Operating system
  - Typically no OS, but instead simple scheduler (or even just interrupts + main code) (foreground/background system)
  - If OS is used, likely to be a lean RTOS

Target Board - FRDM-KL25Z

- 32-bit Cortex M0+ Processor Core
- Freescale Kinetis MKL25Z128VLK4 processor
  - Extremely low power use
  - 48 MHz max clock
  - On-chip 128 KB ROM, 16 KB RAM
  - Wide range of peripherals, including USB on-the-go
- FRDM-KL25Z board
  - $13 (USD)
  - Peripherals: 3-axis accelerometer, RGB LED, capacitive touch slider
  - Expansion ports are compatible with Arduino shield ecosystem – endless opportunities, low-cost hardware
  - mbed.org enabled - online software development toolchain, reusable code



The End!

- Thank You!
- Lets go onto the next adventure!