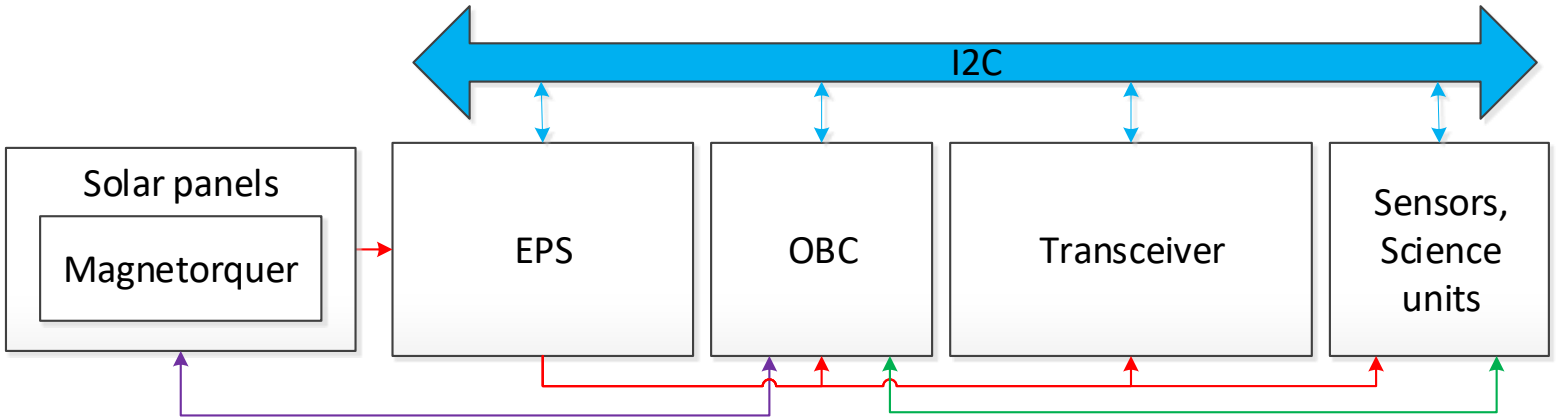
САМАРСКИЙ УНИВЕРСИТЕТ
SAMARA UNIVERSITY

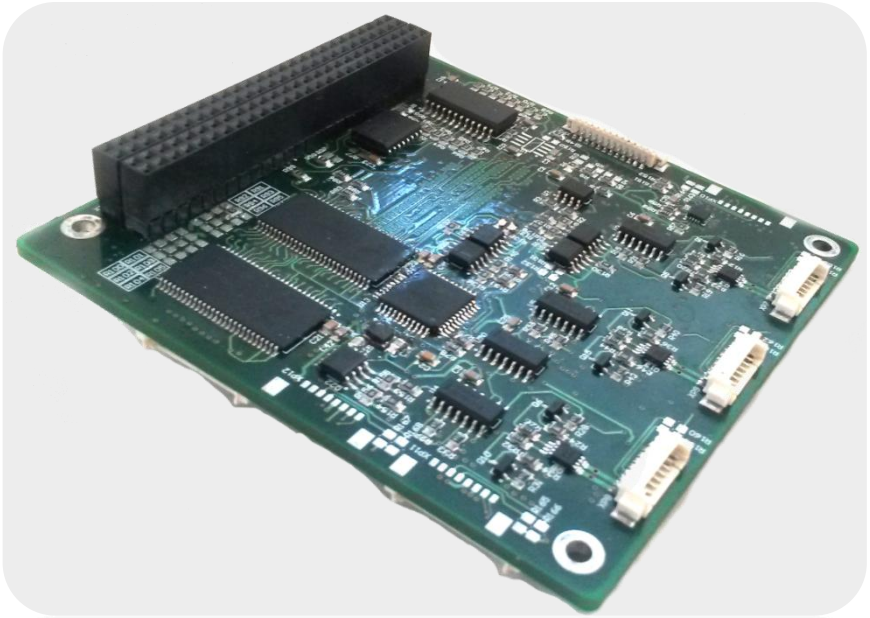
Software development for nanosatellite onboard computers

Dmitriy Kornilin

2021

ONBOARD COMPUTER (OBC)





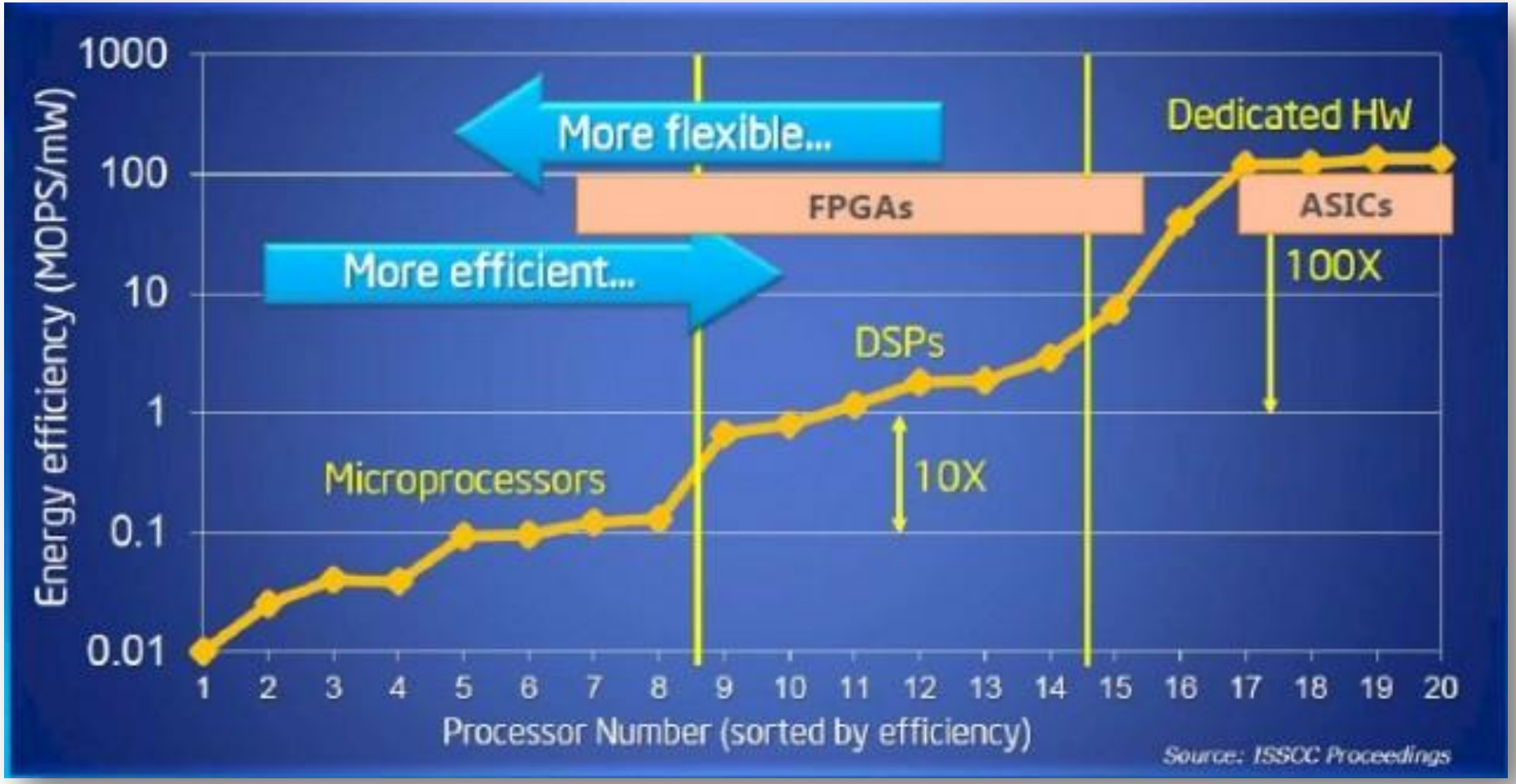
MAIN PROBLEMS

- What tasks has our computer to execute?
- What kind of processor is the most suitable for us?
- What interfaces we need?
- Do we need operating system?
- What language shall we use for programming?
- What data we need to collect and transport to control center?
- How many memory do we need?

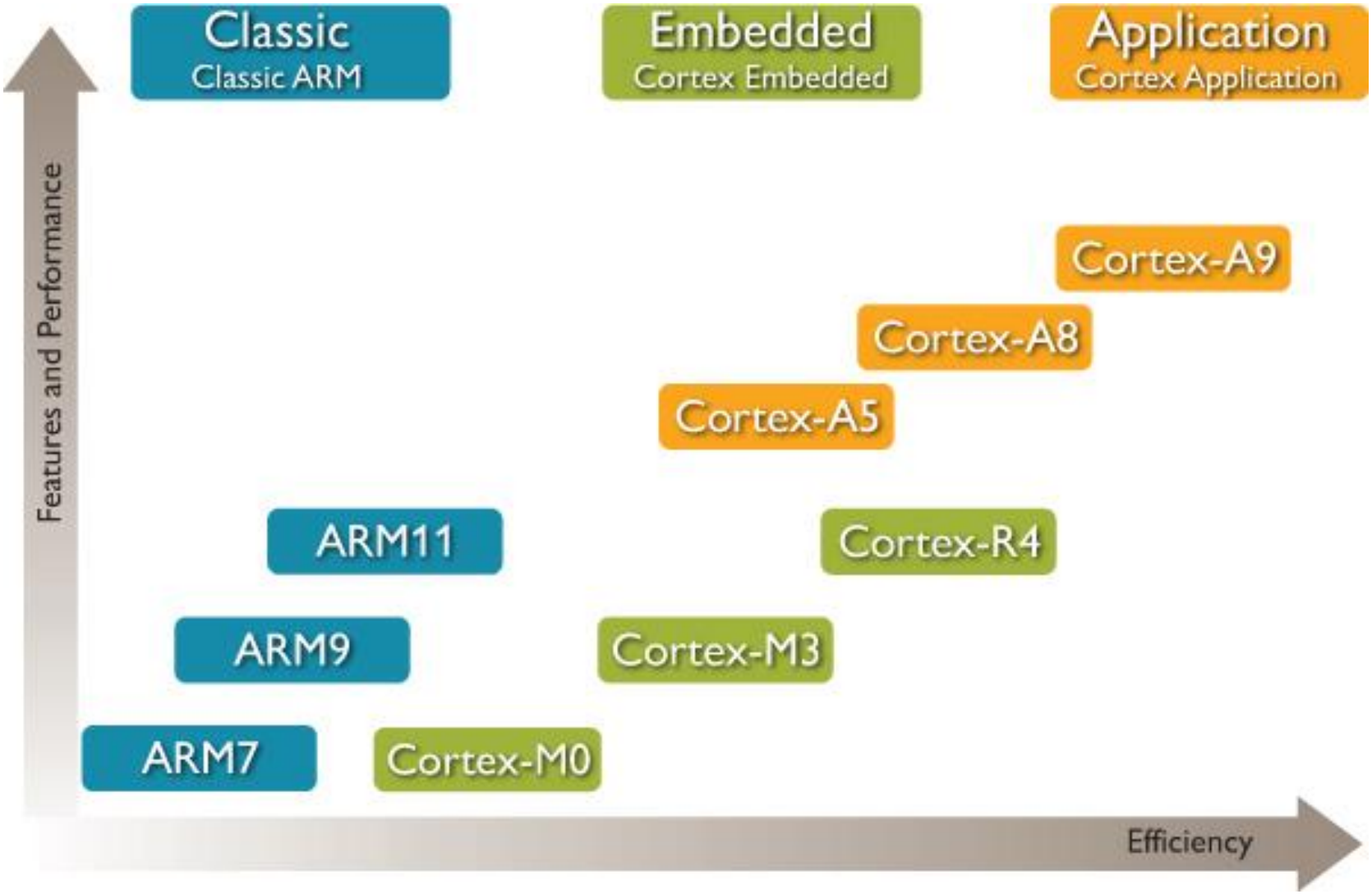
Approaches:

- Ready embedded computer
- Mezzanine boards
- User’s specific solution

HARDWARE CORES

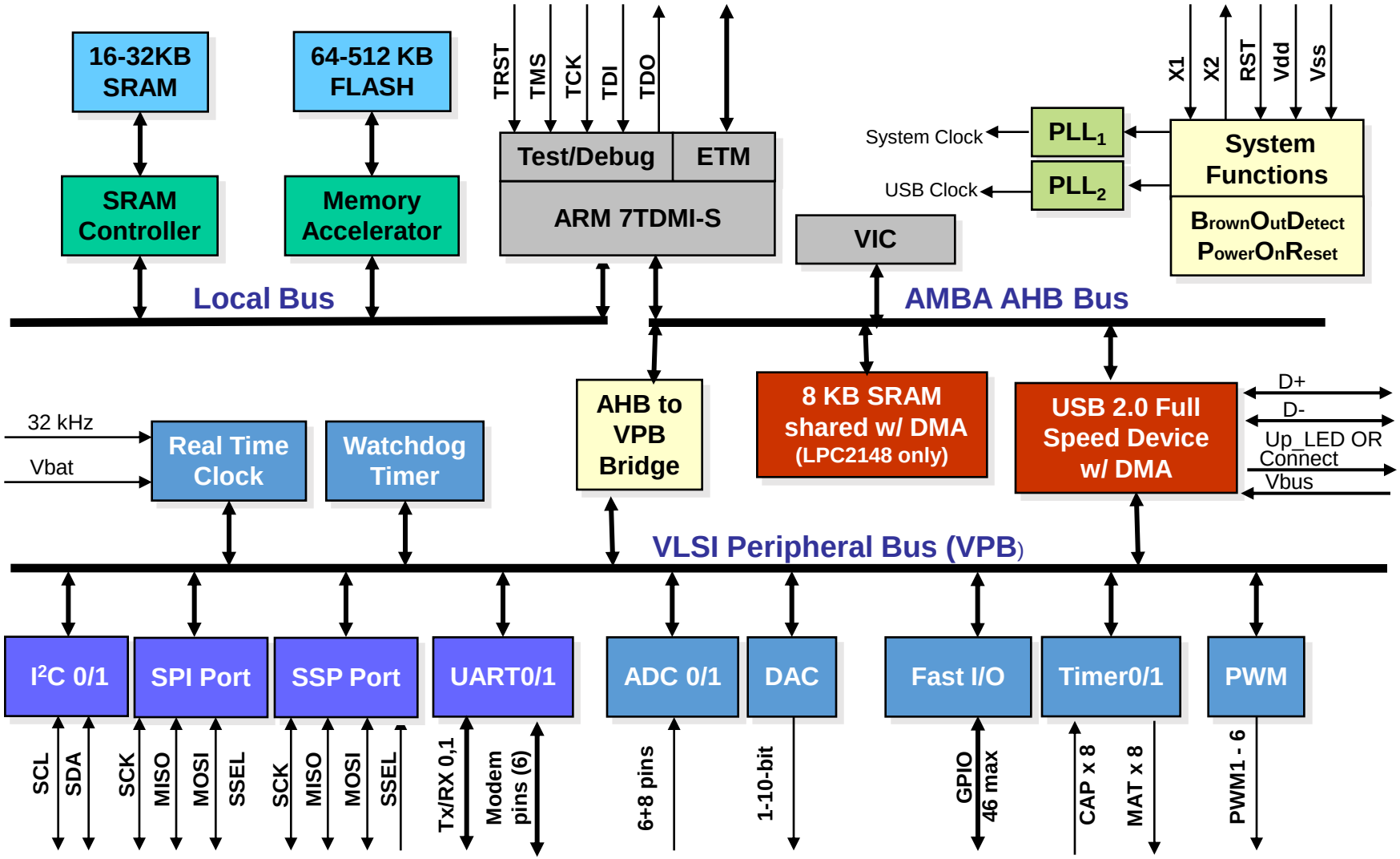


ARM CORES


Classic
Classic ARM
Embedded
Cortex Embedded
Application
Cortex Application
Features and Performance
Cortex-A9
Cortex-A8
Cortex-A5
ARM11
Cortex-R4
ARM9
Cortex-M3
ARM7
Cortex-M0
Efficiency

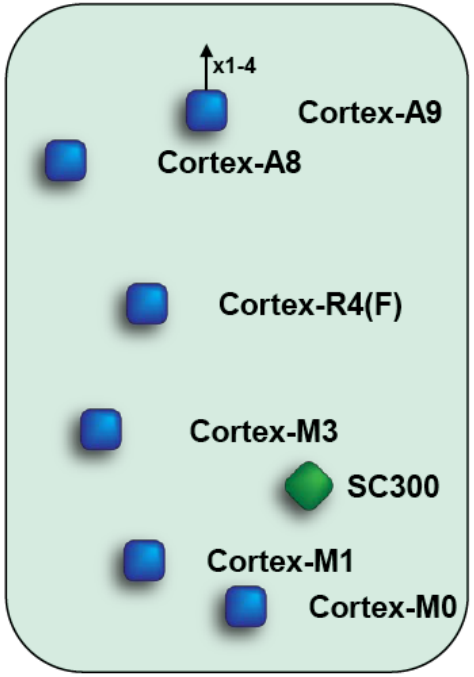
LPC2148	ARM7TDMI
LPC1768	CORTEX-M3
LPC1114	CORTEX-M0
LPC4357	CORTEX-M4

LPC2142/44/46/48 BLOCK DIAGRAM



The ARM Cortex™ Family

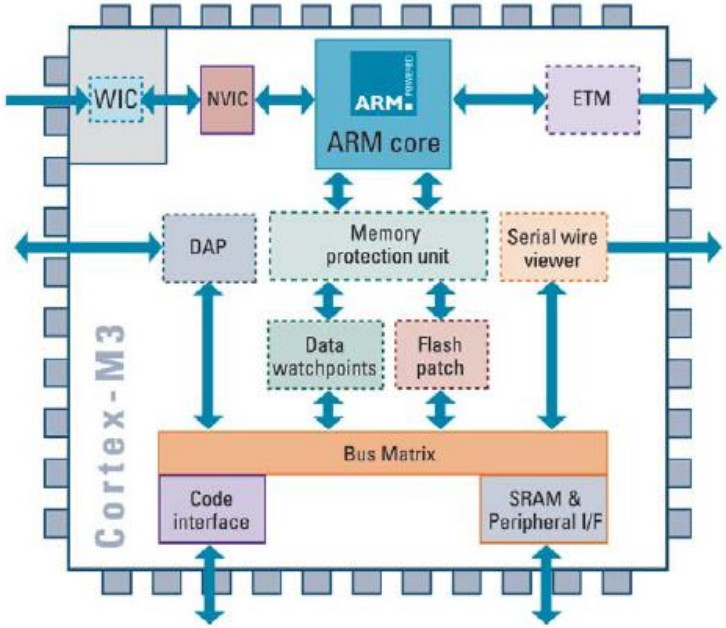
Cortex™
Intelligent Processors by ARM®



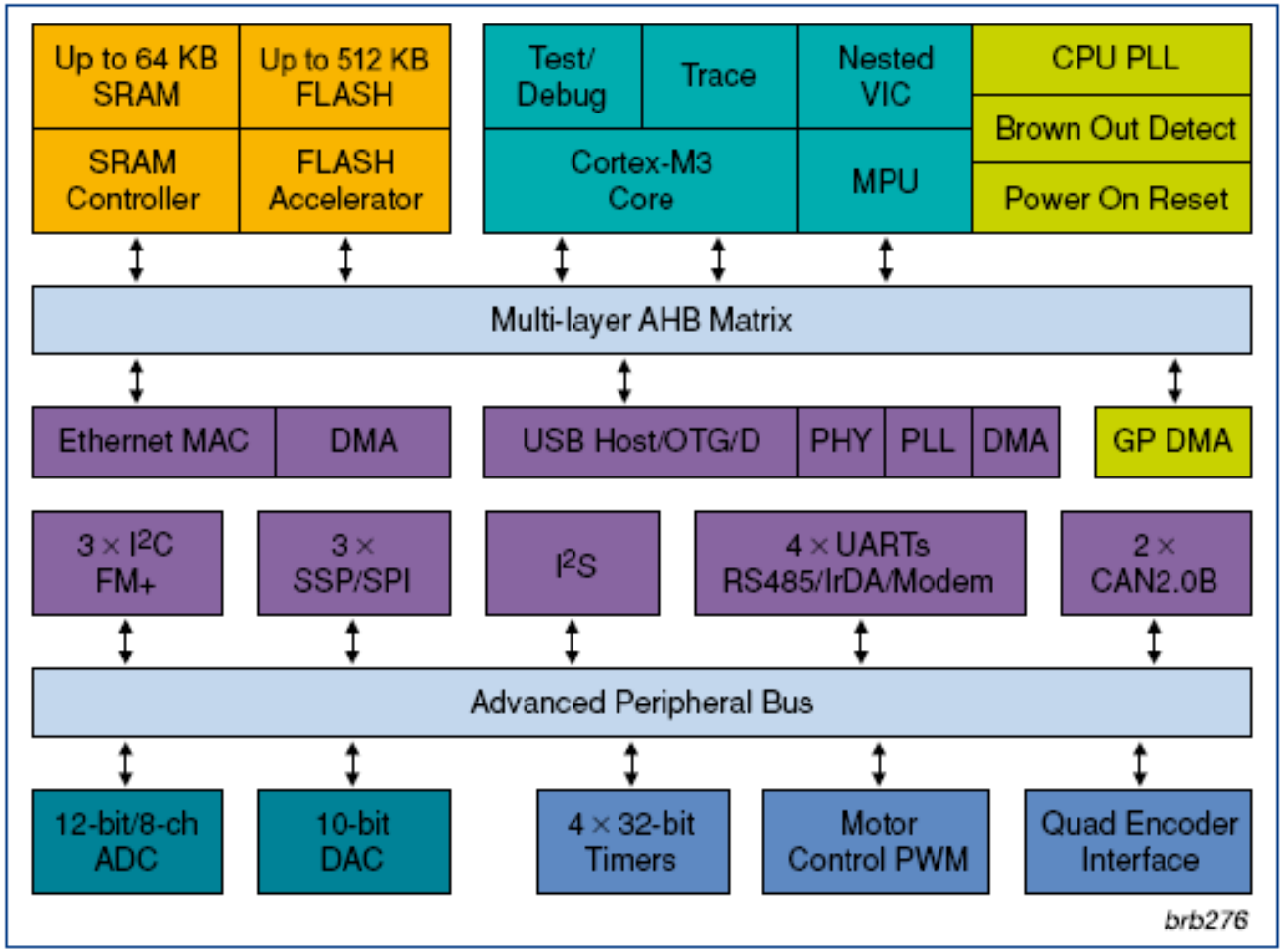
- *ARM Cortex A Series* - ***Applications*** CPUs focused on the execution of complex OS and user applications
- *ARM Cortex R Series* - Deeply embedded processors focused on ***Real-time*** environments
- *ARM Cortex M Series* - ***Microcontroller*** *cores* focused on very cost sensitive, deterministic, interrupt driven environments

Introduction to Cortex-M3 Processor

- **Cortex-M3 Architecture**
  - Harvard bus architecture
  - 3-stage pipeline with branch speculation
  - Integrated bus matrix
  - Configurable nested vectored interrupt controller (NVIC)
  - Advanced configurable debug and trace components
  - Optional components for specific market requirements:
    - Wake-up Interrupt Controller (WIC)*
    - Memory Protection Unit (MPU)
    - Embedded Trace Macrocell (ETM)
    - Fault Robust Interface*



LPC1700 Block Diagram

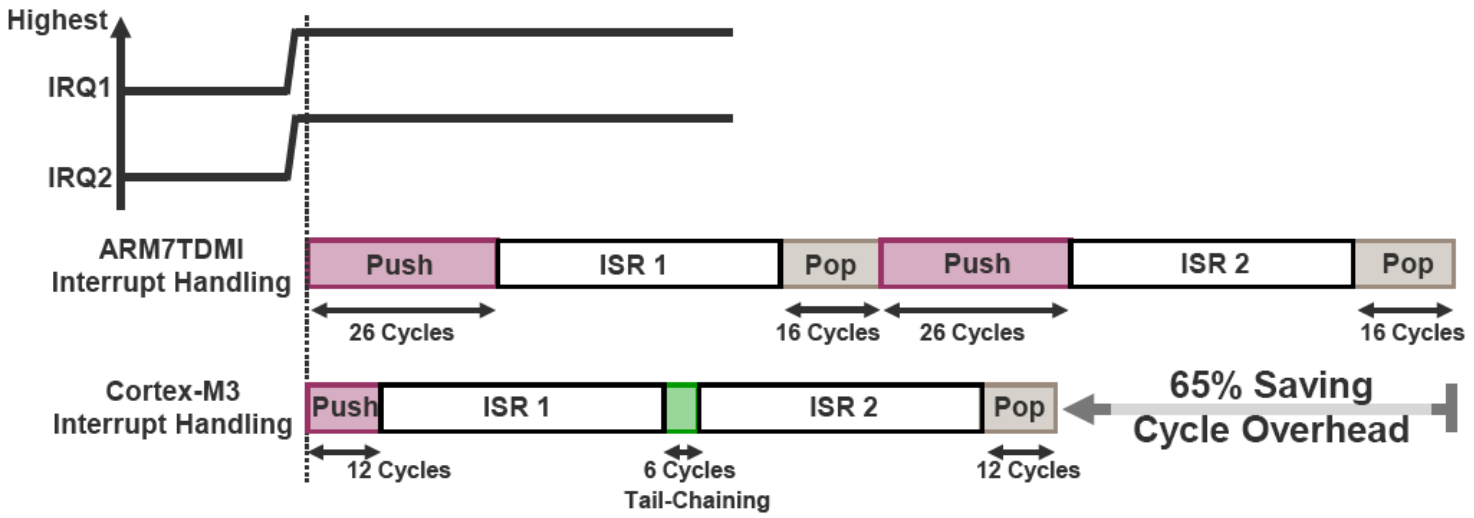


BENEFITS OF CORTEX M3 COMPARED TO ARM7

Table 1. ARM7TDMI-S and Cortex-M3 comparison (100MHz frequency on TSMC 0.18G)

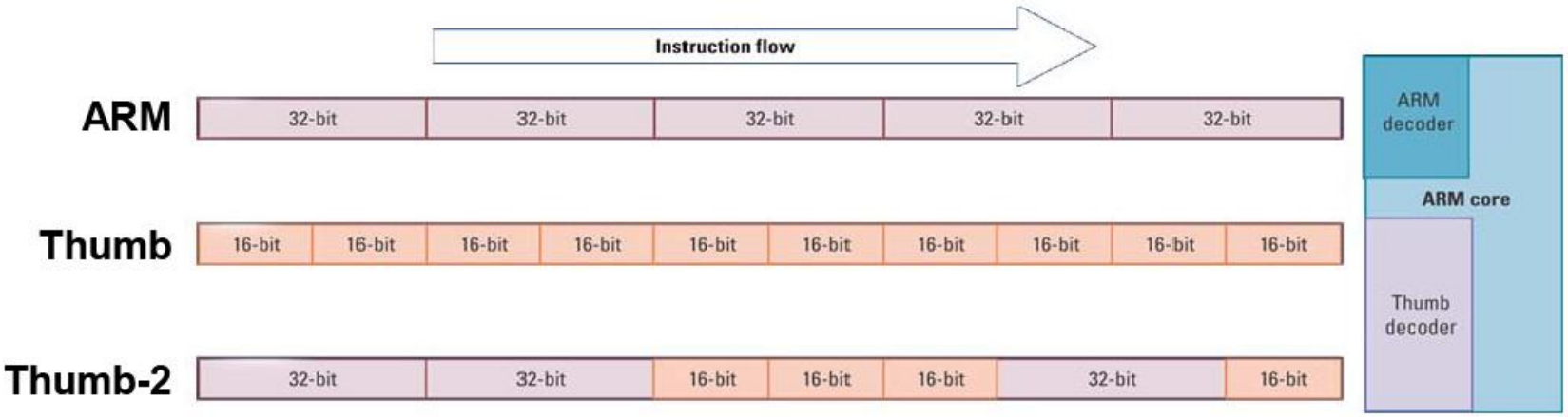
Features	ARM7TDMI-S	Cortex-M3
Architecture	ARMv4T (von Neumann)	ARMv7-M (Harvard)
ISA Support	Thumb / ARM	Thumb / Thumb-2
Pipeline	3-Stage	3-Stage + branch speculation
Interrupts	FIQ / IRQ	NMI + 1 to 240 Physical Interrupts
Interrupt Latency	24-42 Cycles	12 Cycles
Sleep Modes	None	Integrated
Memory Protection	None	8 region Memory Protection Unit
Dhrystone	0.95 DMIPS/MHz (ARM mode)	1.25 DMIPS/MHz
Power Consumption	0.28mW/MHz	0.19mW/MHz
Area	0.62mm2 (Core Only)	0.86mm2 (Core & Peripherals)*

Interrupt Response – Tail Chaining



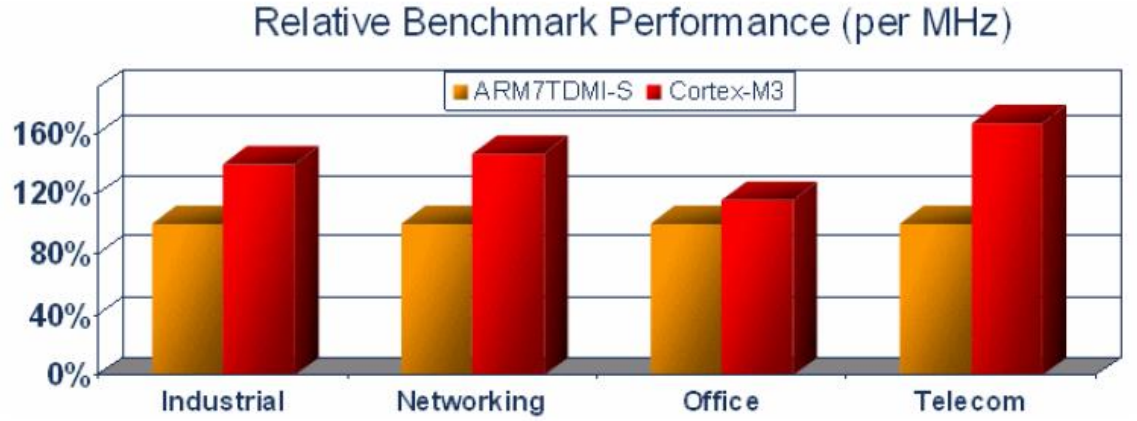
ARM7TDMI	Cortex-M3
▪26 cycles from IRQ1 to ISR1 (up to 42 cycles if in LSM) ▪ 42 cycles from ISR1 exit to ISR2 entry ▪ 16 cycles to return from ISR2	▪ 12 cycles from IRQ1 to ISR1 (Interruptible/Continual LSM) ▪ 6 cycles from ISR1 exit to ISR2 entry ▪ 12 cycles to return from ISR2

Thumb + additional Thumb + 32- bit commands

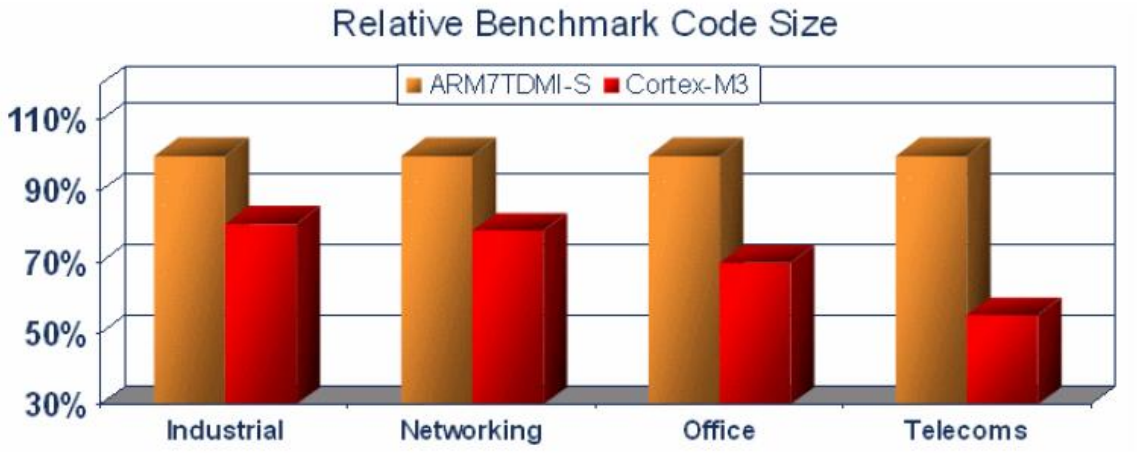


ARM7 AND CORTEX M3 PERFOMANCE

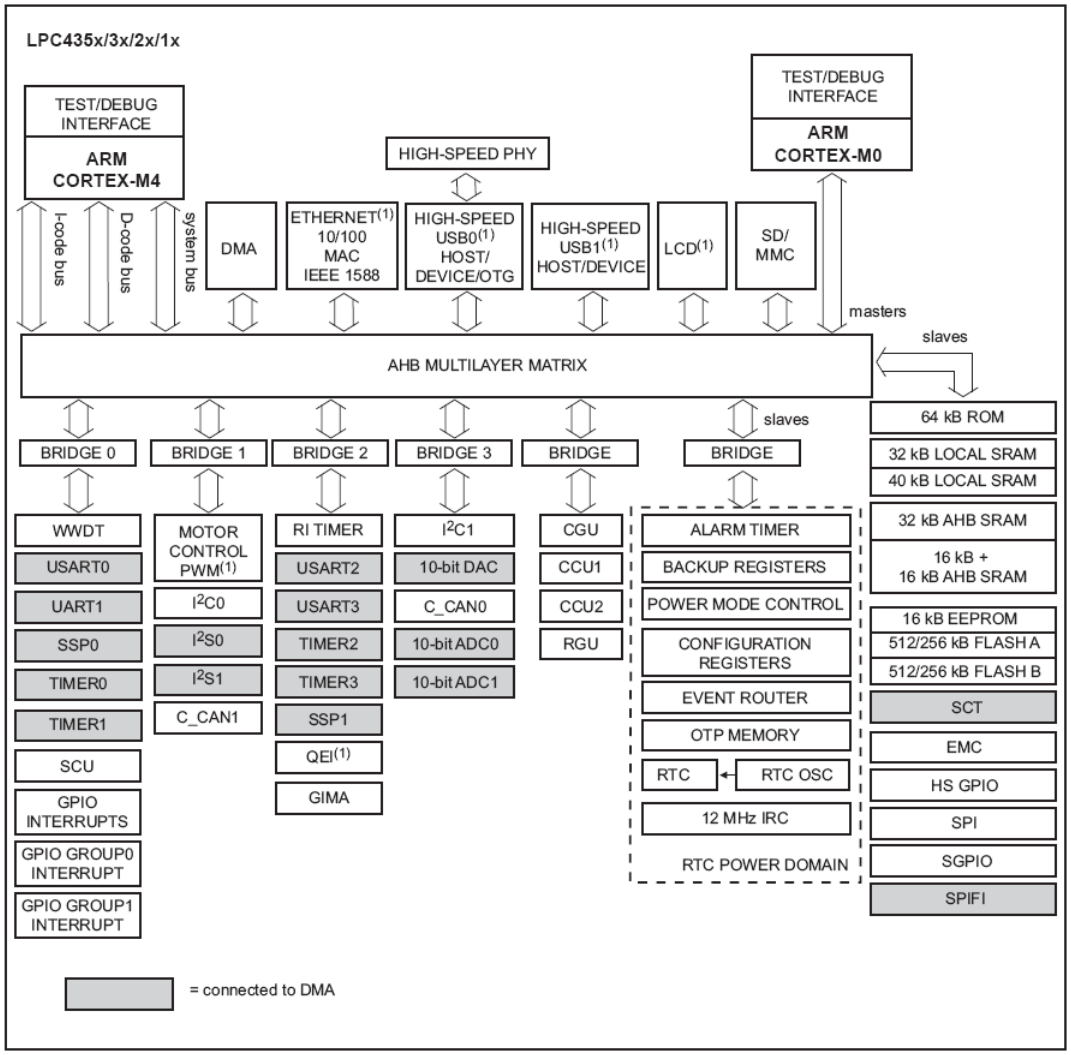
Relative performance for ARM7TDMI-S (ARM) and Cortex-M3 (Thumb-2)

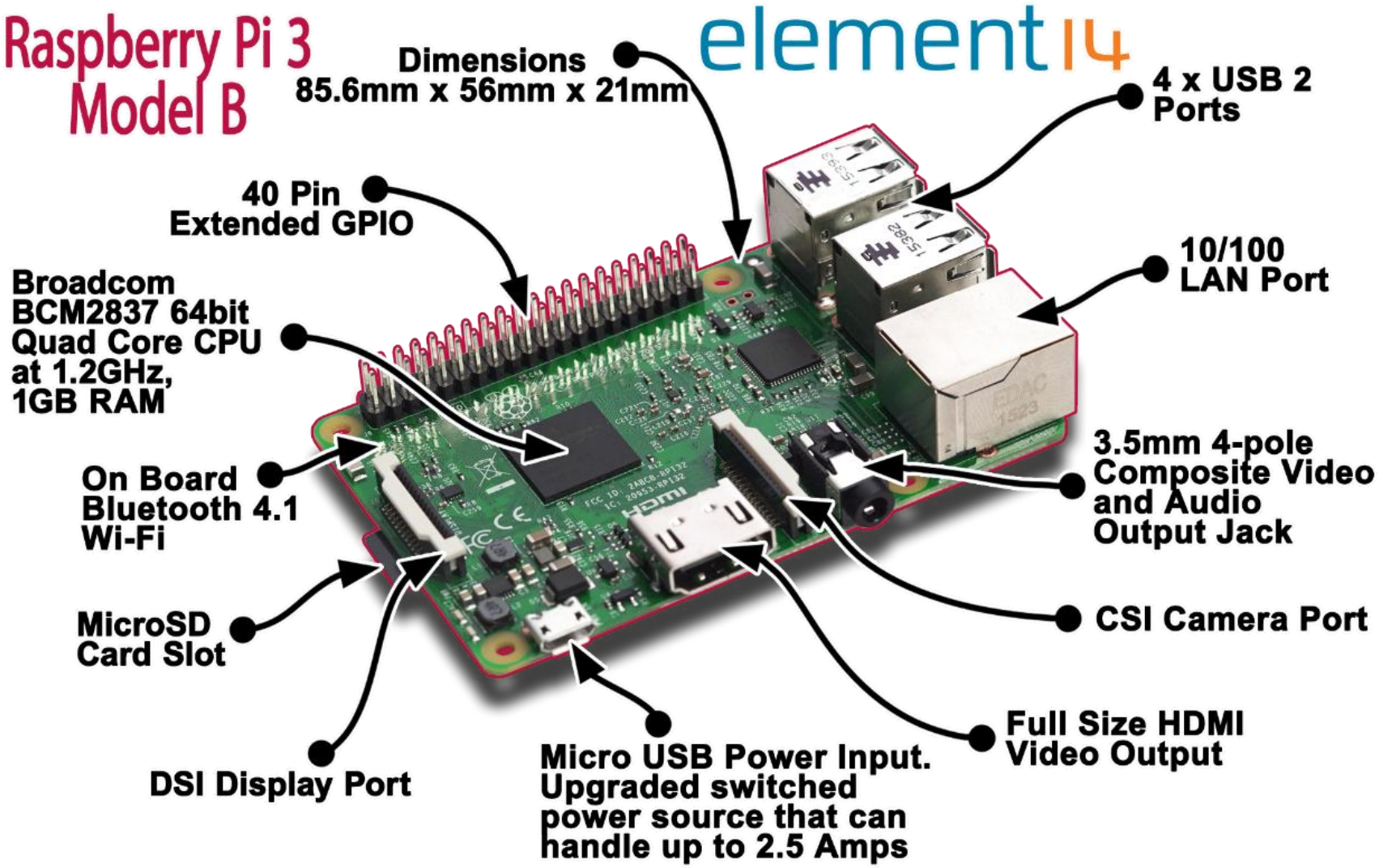


Relative code size for ARM7TDMI-S (ARM) and Cortex-M3 (Thumb-2)

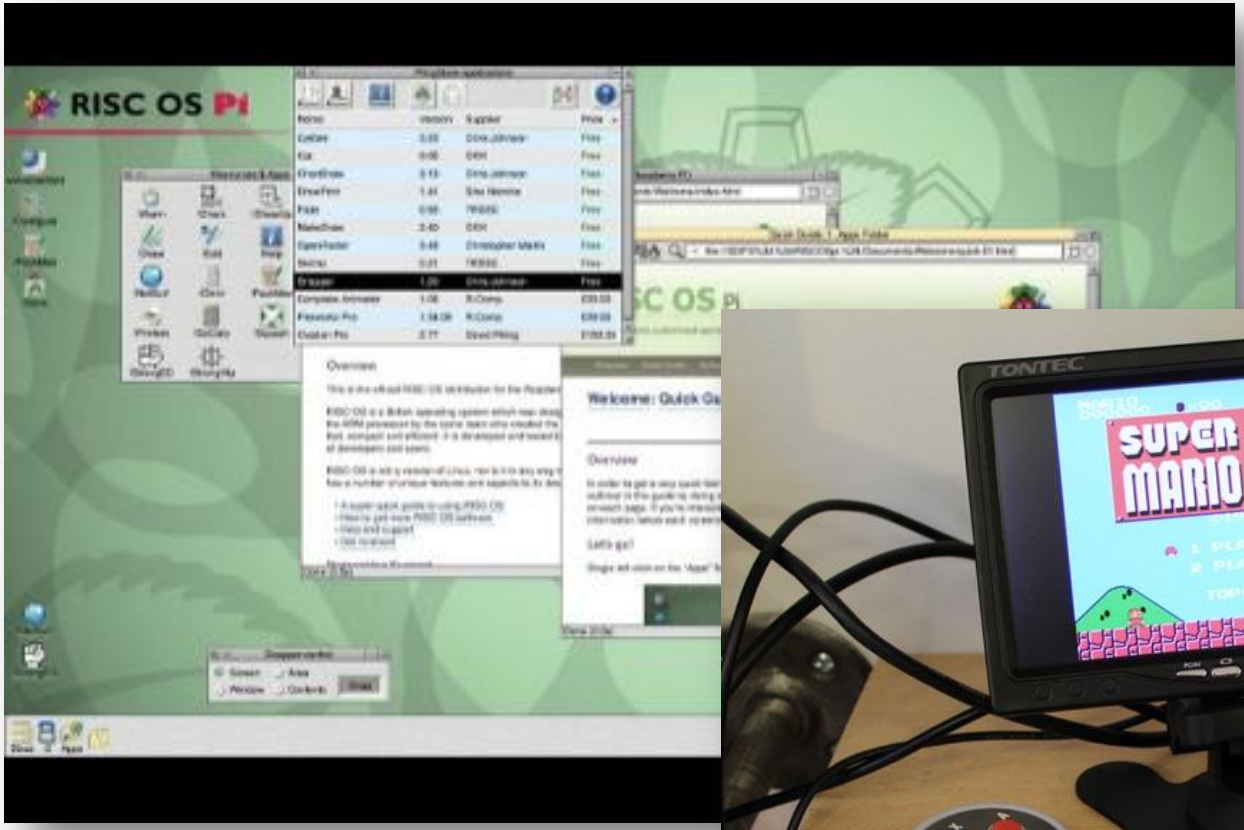


MULTICORE CORTEX M4+M0




Raspberry Pi 3
Model B
element14
Dimensions
85.6mm x 56mm x 21mm
4 x USB 2
Ports
40 Pin
Extended GPIO
10/100
LAN Port
Broadcom
BCM2837 64bit
Quad Core CPU
at 1.2GHz,
1GB RAM
On Board
Bluetooth 4.1
Wi-Fi
3.5mm 4-pole
Composite Video
and Audio
Output Jack
MicroSD
Card Slot
CSI Camera Port
DSI Display Port
Micro USB Power Input.
Upgraded switched
power source that can
handle up to 2.5 Amps
Full Size HDMI
Video Output

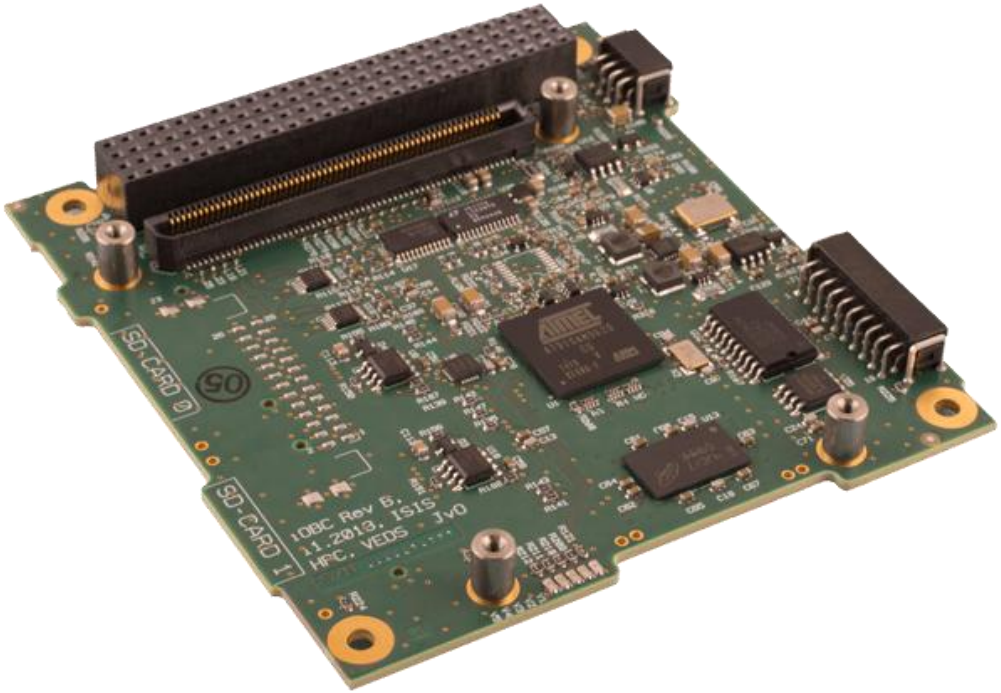
GENERAL SOLUTION



NANOMIND A712D

- ARM7 processor - 8-40 MHz
- 2 MB RAM, 4 MB Code storage, 4 MB data flash
- CAN and I2C interfaces
- RTC - real time clock w/backup power keeps time 30-60 minutes without external power
- FreeRTOS OS and driver library included
- MicroSD socket for up to 2 GB storage
- On-board magnetometer
- 3 PWM bidirectional output 3.3-5 V/±3 A (Compatible with NanoPower Solar P110U)
- Interface to 6-analog inputs (e.g. sun sensors) (Compatible with NanoPower Solar P110U)
- SPI interface to e.g. gyroscopes

IOBC



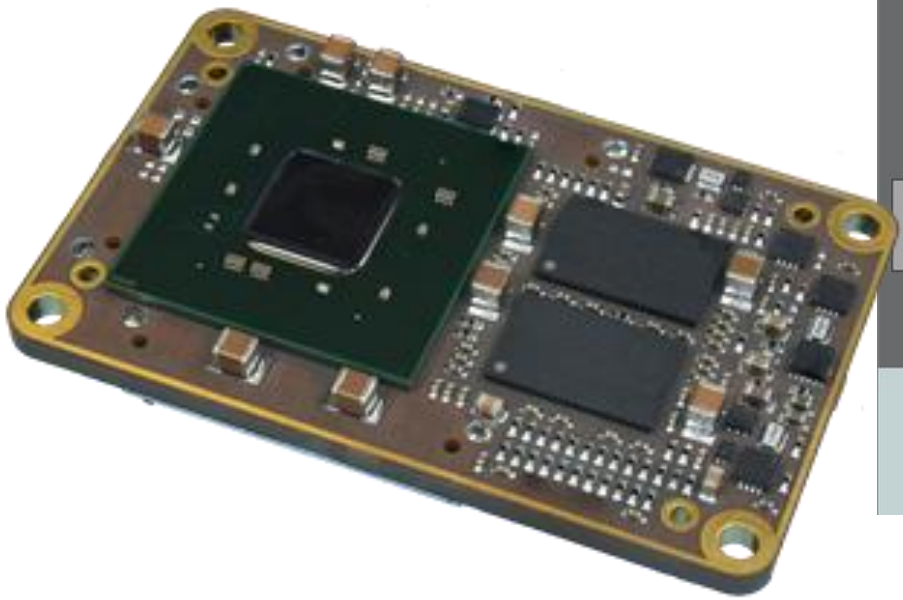
- Processor: 400MHz 32-bit ARM9 processor
- Volatile Memory: 64MB RAM
- Data Storage: 2x any size standard SD cards
- Code Storage: 1MB NOR Flash
- FRAM: 256KB

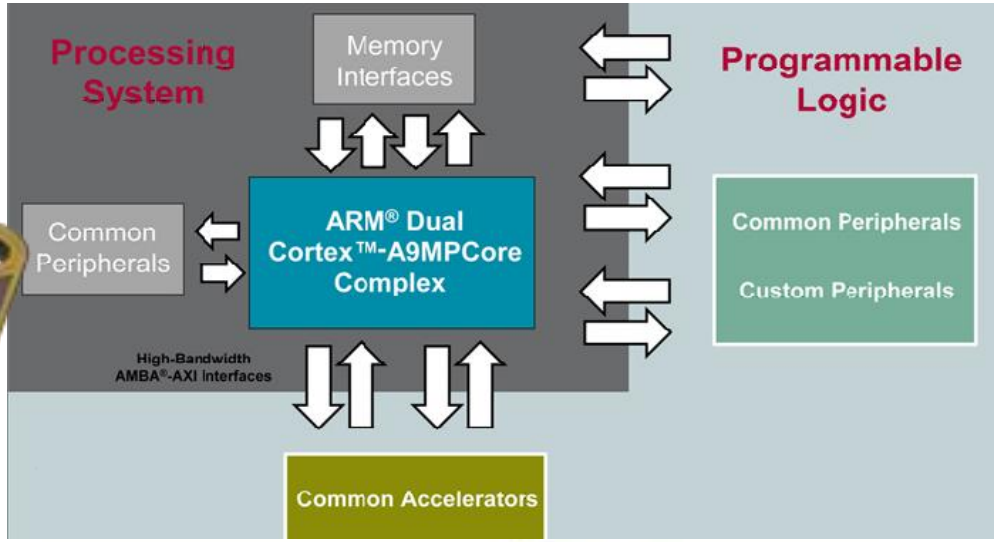
- Power Consumption: 400mW average (550 mW max @3.3V)

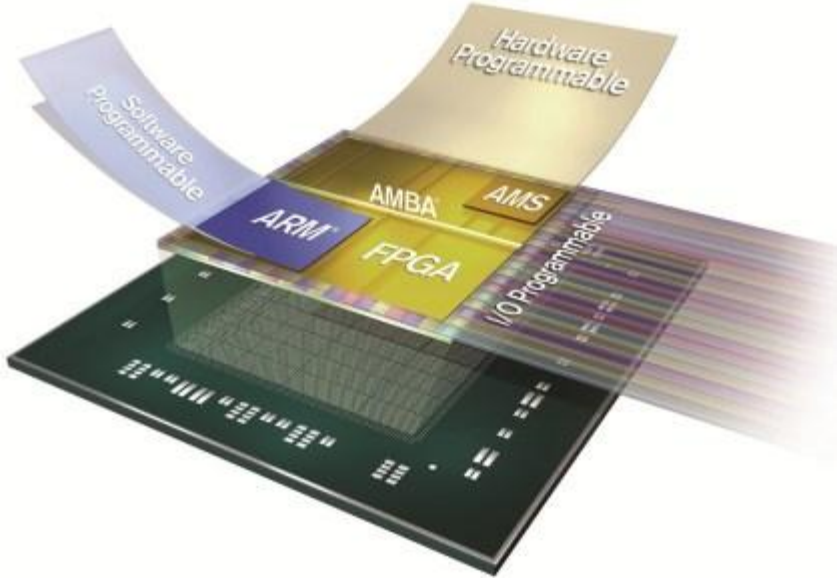
Interface:

- I2C (master or slave, up to 400 Kbits/s on fast mode)
- SPI: up to 8 slaves, up to 10 Mbit/s
- UART: 2x RS232 / 1x RS232 + 1x RS485
- ADC: 8 channel, 10-bit
- PWM: 6 channel
- GPIO: up to 27
- 1x USB host
- 1x USB device
- Image Sensor Interface for directly interfacing CMOS Image Sensors
- JTAG for programming
- Debug port: UART for console user-interface
- 4x LEDs (Engineering Model only) to support testing and debugging

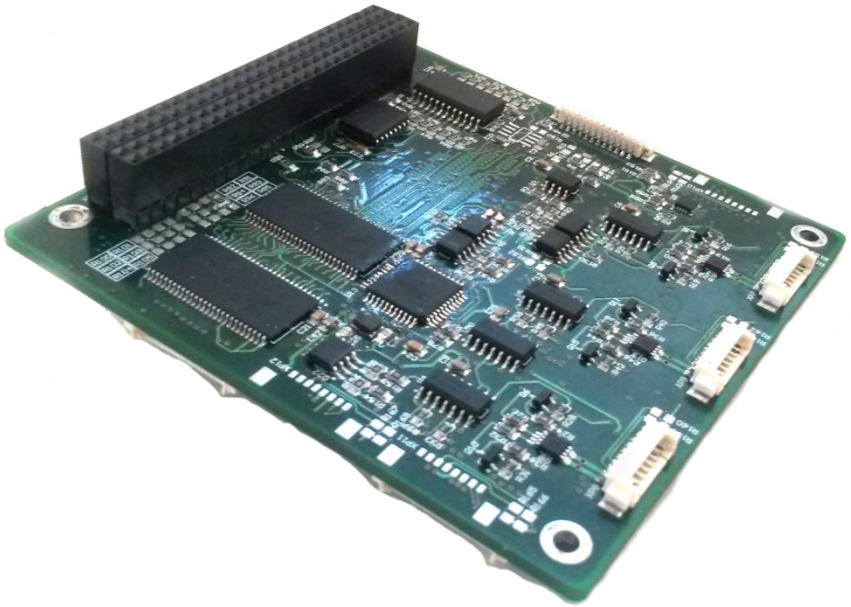
NANOMIND Z7000





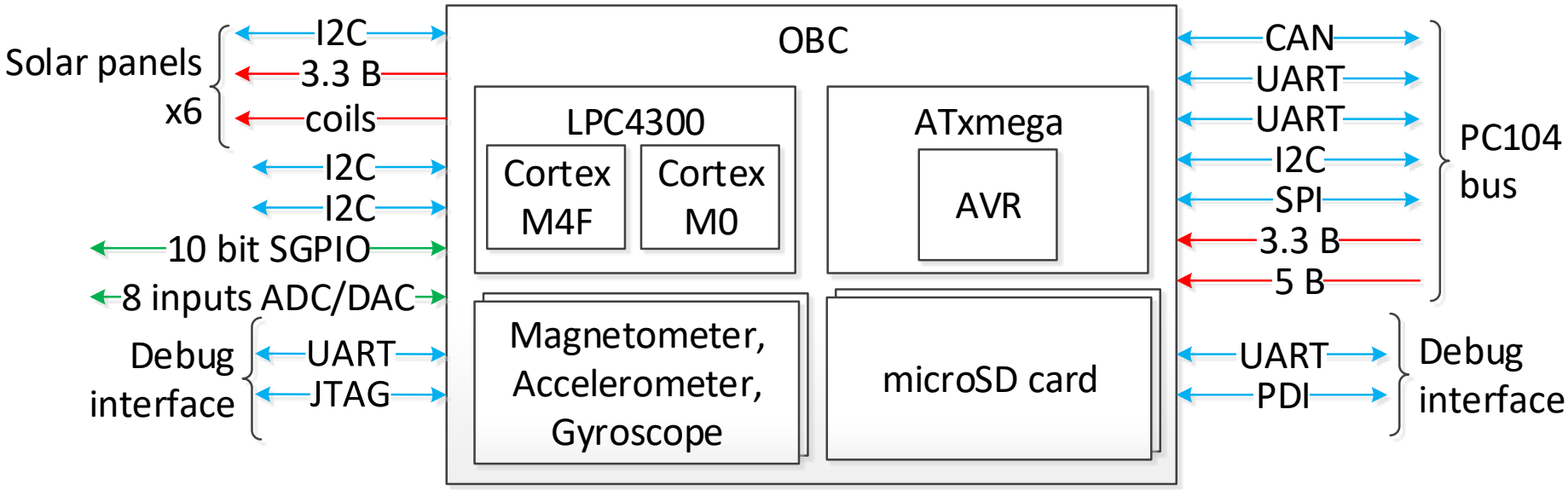


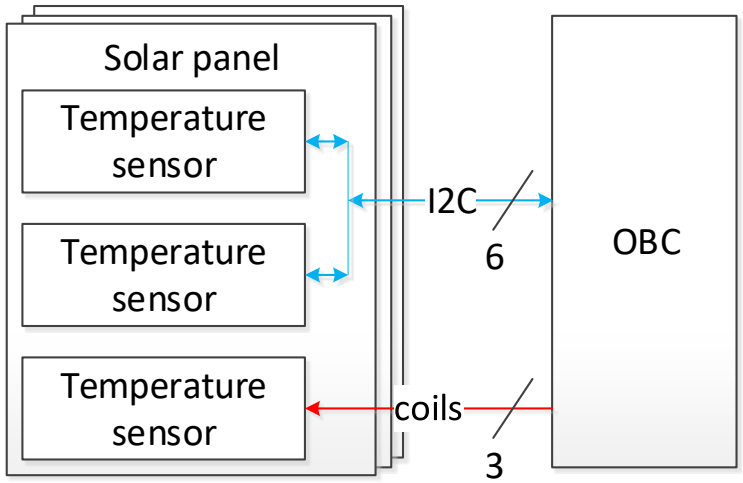
- Xilinx Zynq 7030 Programmable SoC
- Dual ARM Cortex A9 MPCore up to 800 MHZ
- 1 GB DDR3 RAM and 4 GB storage (32 GB ns option)
- FPGA module – 125k logic cells



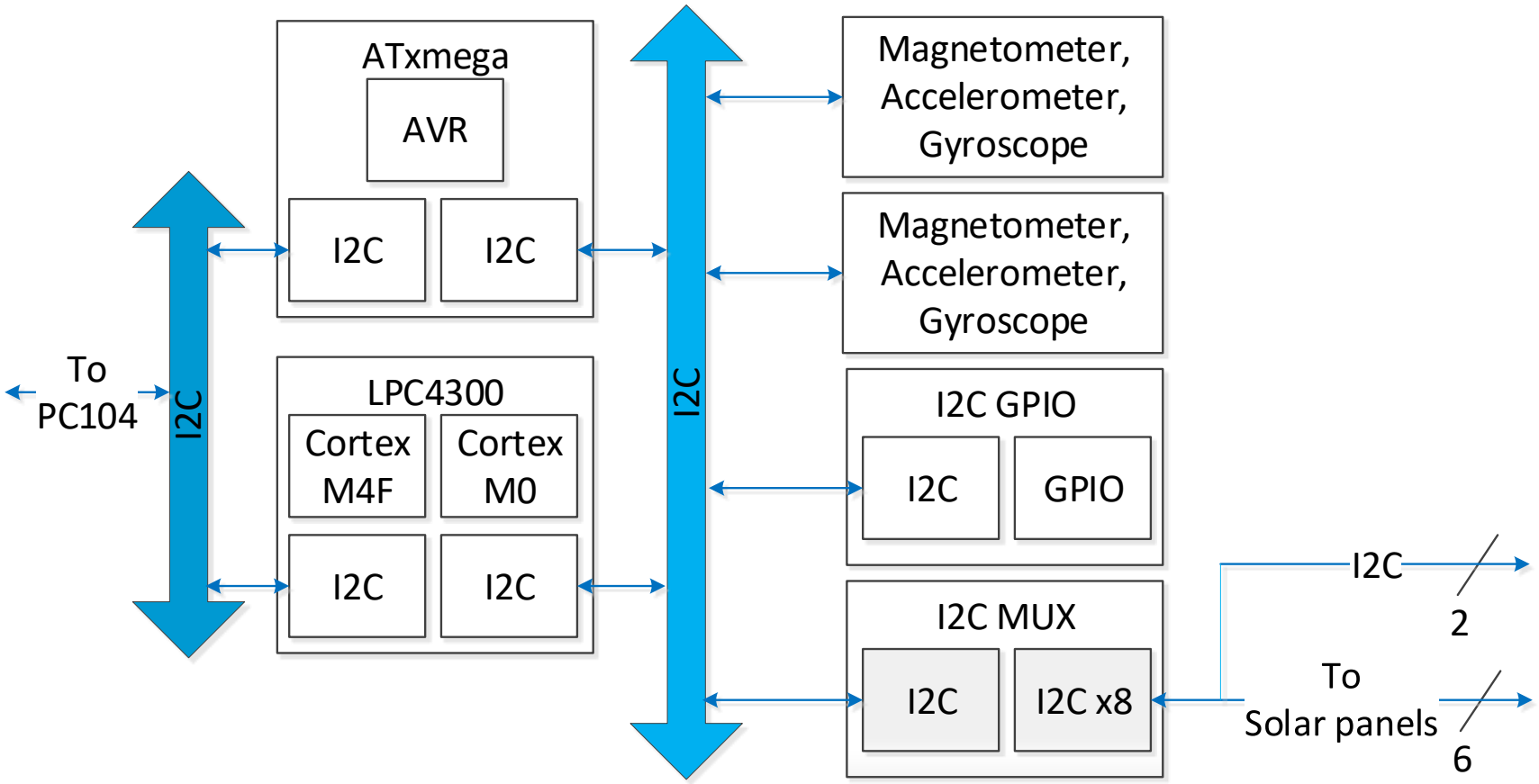
- Two independent microcontrollers
- Three cores: Cortex M4F, M0, ATxmega
- Operating frequency up to 204 MHz
- Two shared microSD cards up to 32 GB each
- Two independent set of 3-axis magnetometer, gyroscope and accelerometer
- RTC with battery back up power
- Three PWM channels for magnetorquers

SSAU OBC

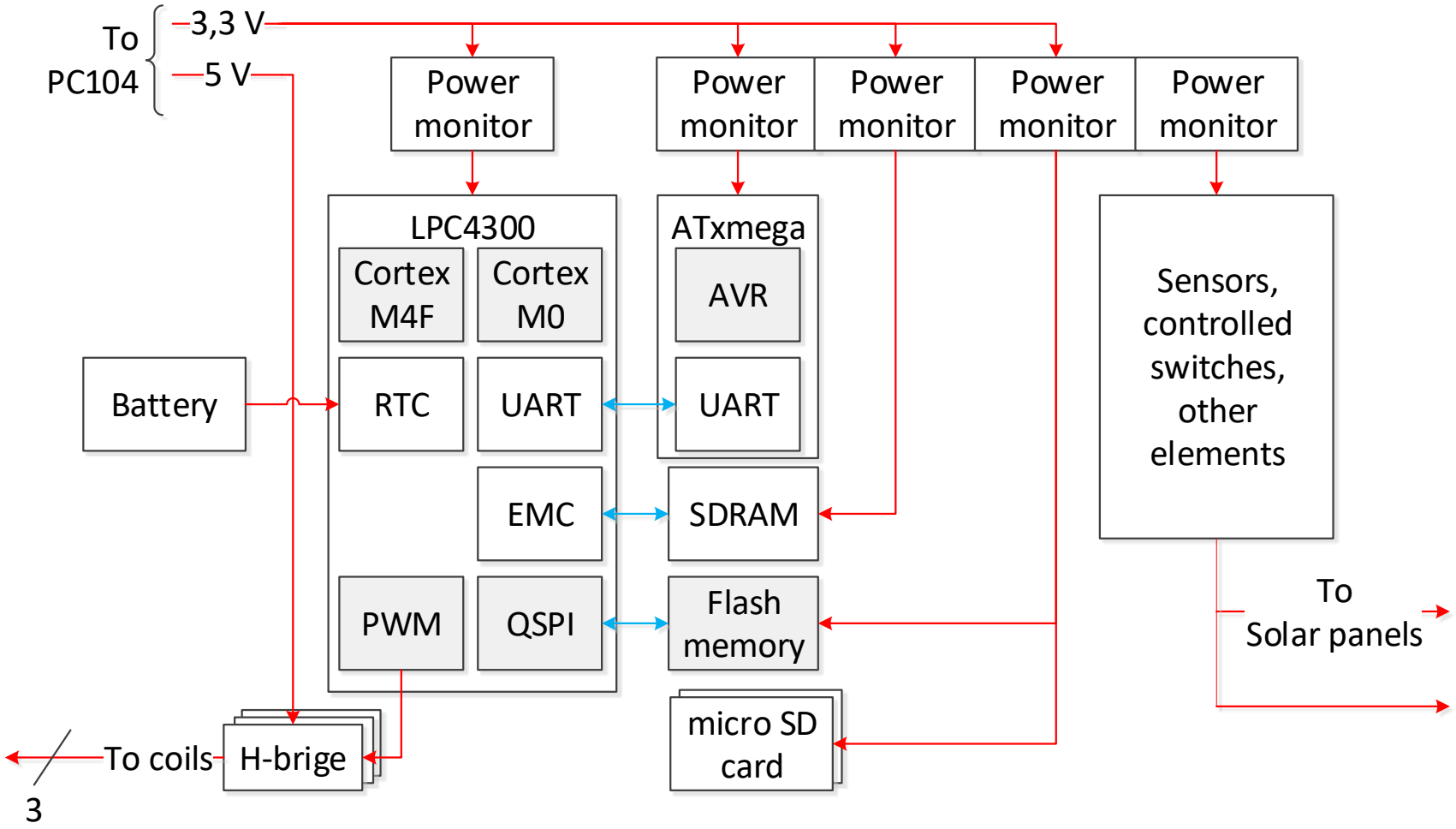

Solar panels x6
I2C
3.3 B
coils
I2C
I2C
10 bit SGPIO
8 inputs ADC/DAC
Debug interface
UART
JTAG
OBC
LPC4300
Cortex M4F
Cortex M0
ATxmega
AVR
Magnetometer, Accelerometer, Gyroscope
microSD card
CAN
UART
UART
I2C
SPI
3.3 B
5 B
PC104 bus
UART
PDI
Debug interface


Solar panel
Temperature sensor
Temperature sensor
Temperature sensor
I2C
6
coils
3
OBC

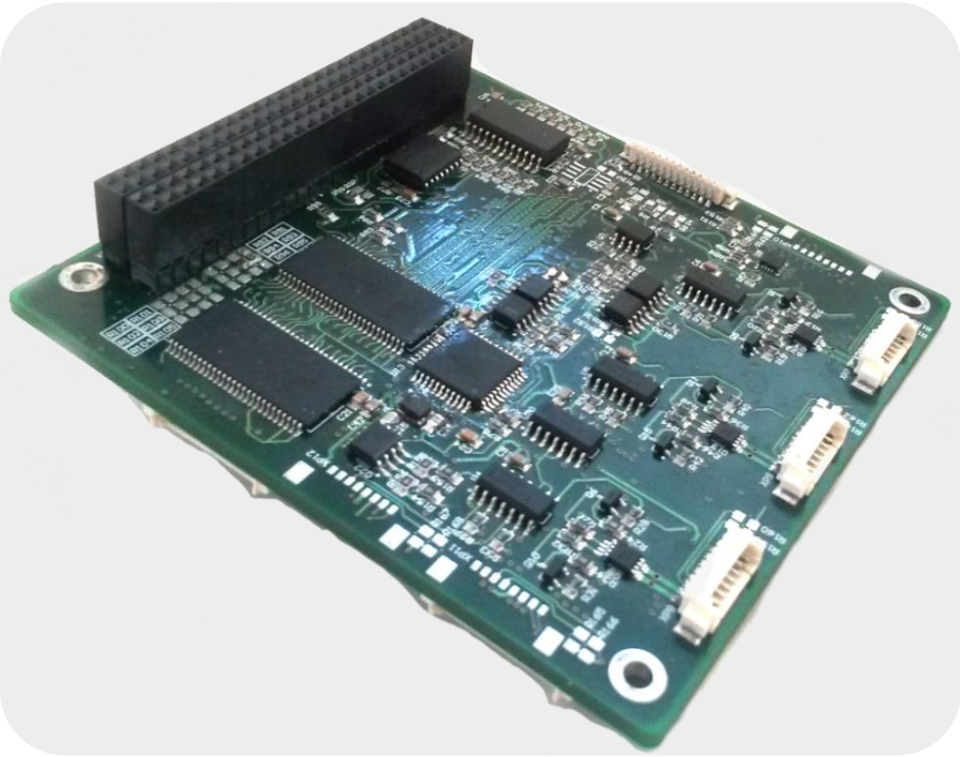
SSAU OBC STRUCTURE

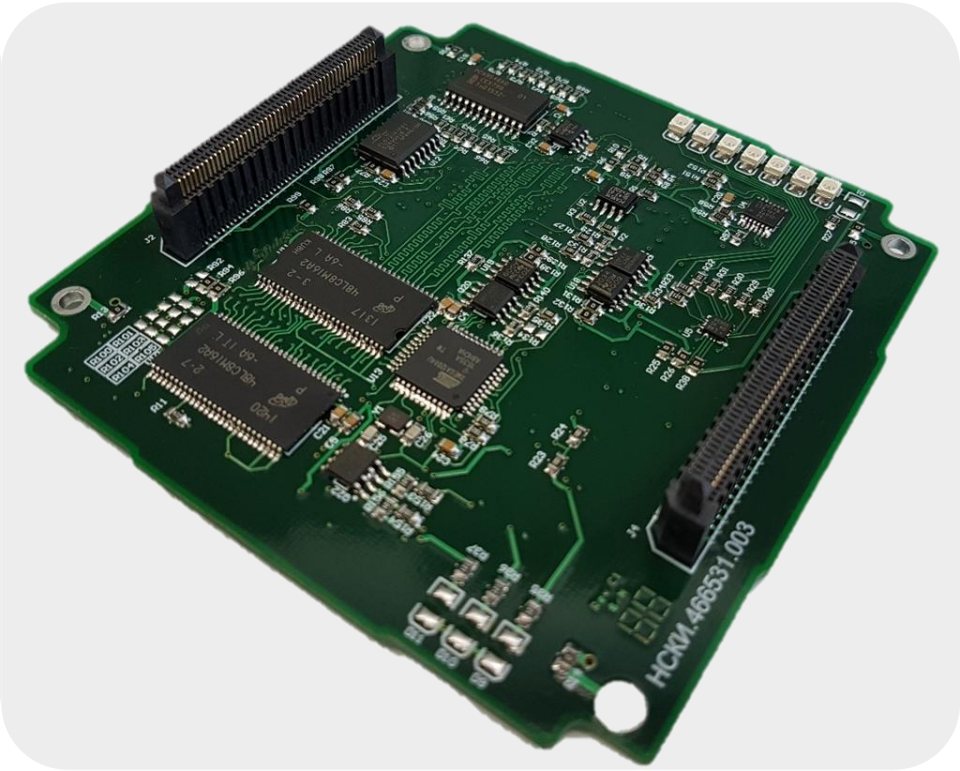


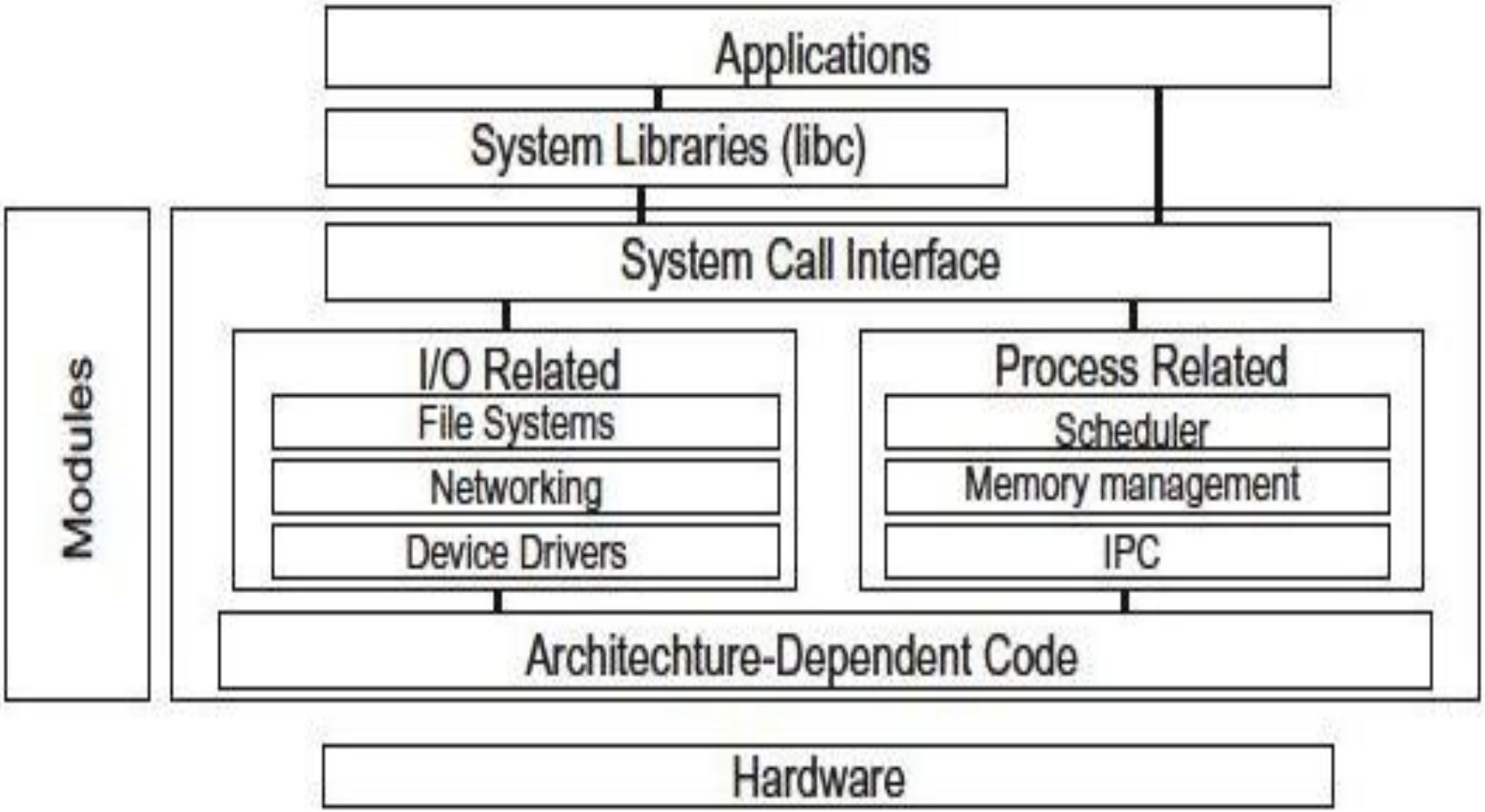
SSAU OBC POWER SUPPLY



SSAU OBC INTERFACES

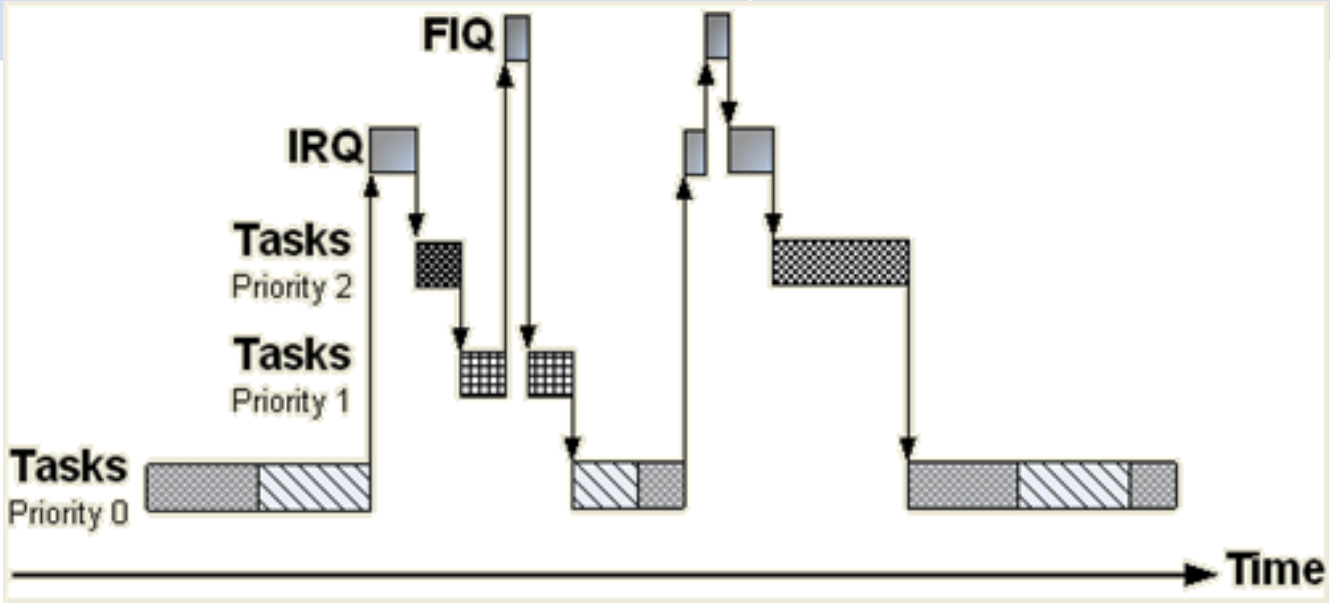


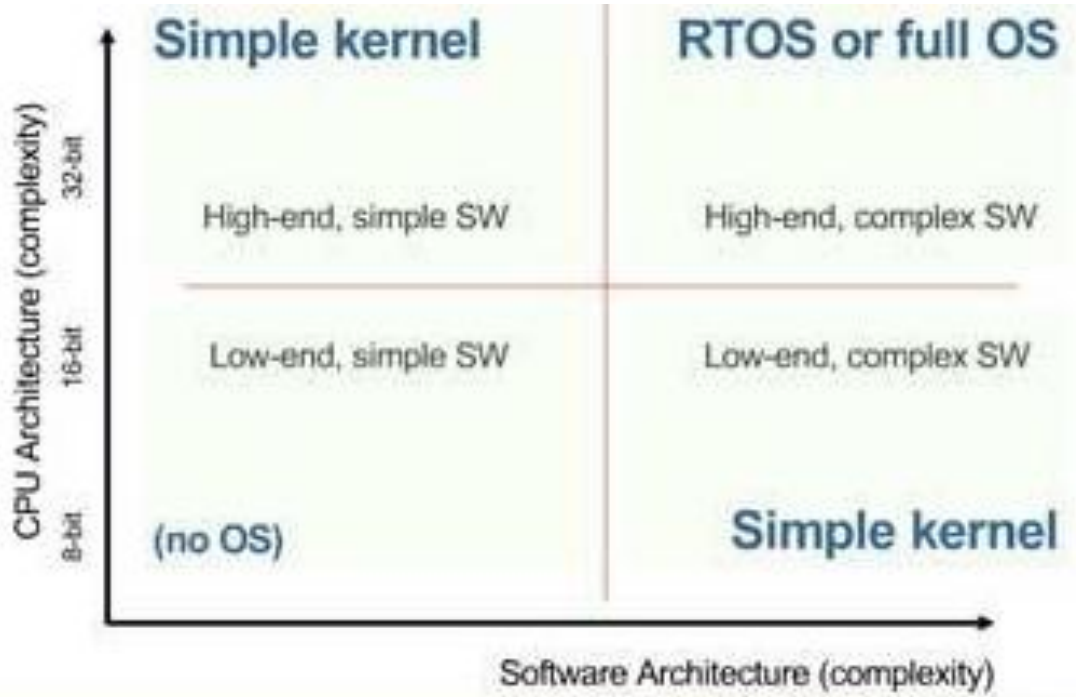



Applications
System Libraries (libc)
Modules
System Call Interface
I/O Related
File Systems
Networking
Device Drivers
Process Related
Scheduler
Memory management
IPC
Architechture-Dependent Code
Hardware

Operating systems

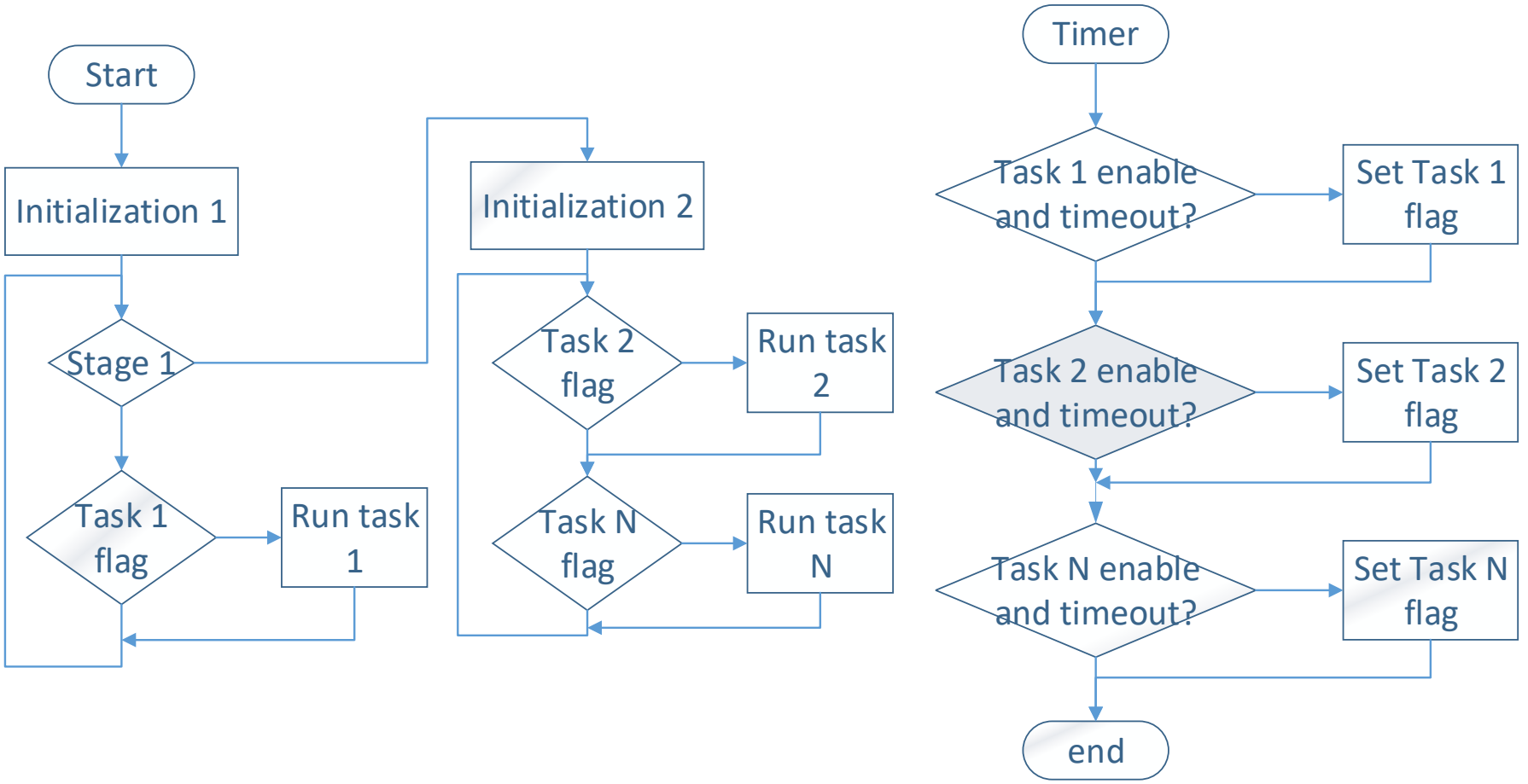
General purpose operating systems (GPOS)	Real-time operating systems (RTOS)
Windows, Unix, Linux, etc.	QNX, FreeRTOS, Salvo, uC/OS etc.
Designed to do many things but are not designed to offer strict guarantees of: • availability (how often the system responds to requests in a timely manner) • reliability (how often these responses are correct)	Engineered to guarantee: • availability • reliability





Advantages	Drawbacks
Developer can use built-in features	Code size increases
Less software dependency on hardware	Efficiency can decrease
Multitasking and protection	Dependency on OS principles

SIMPLE KERNEL



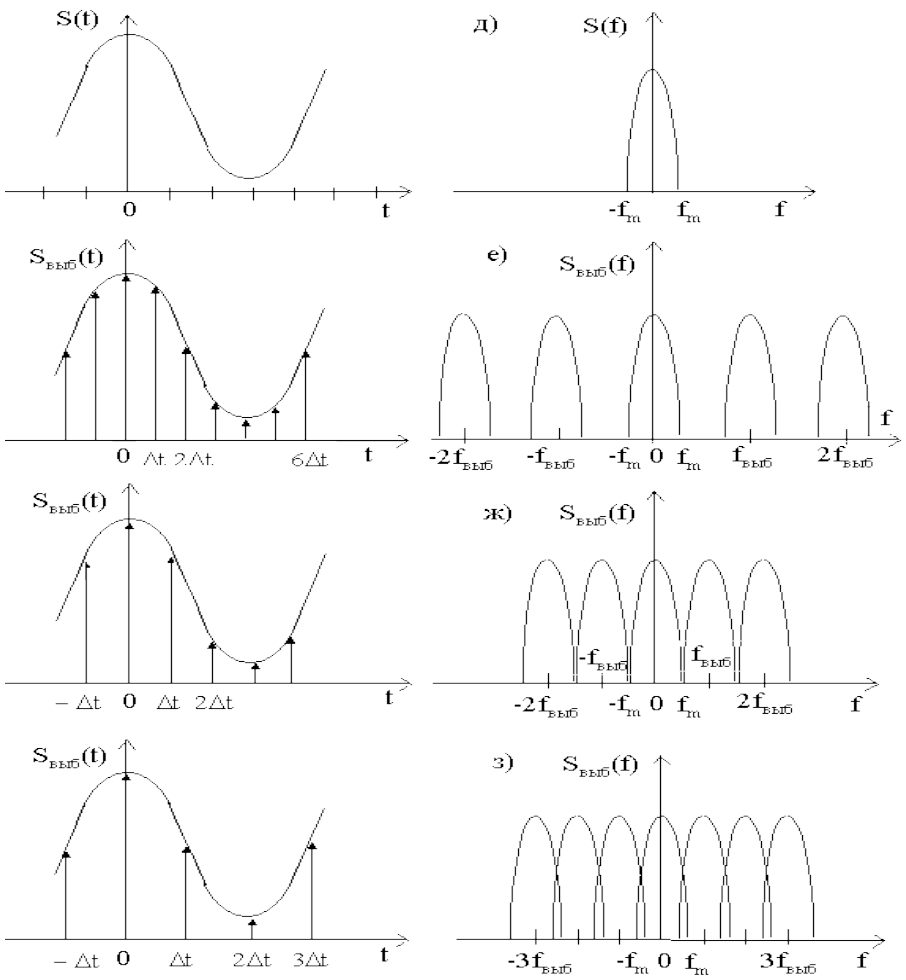
DATA BUDGET

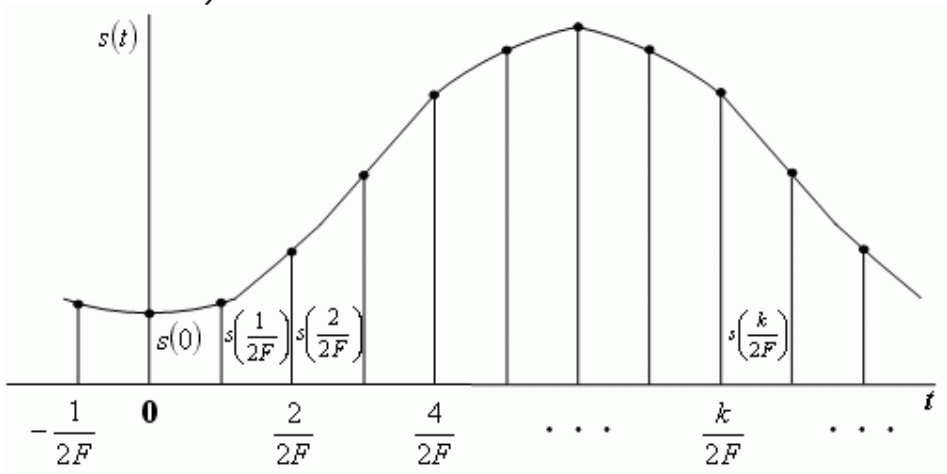
Item	Telemetry type	Quantity	Sampling rate (Hz)	Word size (bits)	Required data rate (bps)
EPS	voltage	3	0.02	16	0.96
	current	3	0.02	8	0.48
OBC	status	1	0.02	16	0.32
	time	1	4	32	128
ADCS	magnetometer	3	4	10	120
	accelerometer	3	4	16	192
	gyroscope	3	4	16	192
	Light sensor, RGBW	4x6	4	8	768
				PCM Data Rate	1 401.76

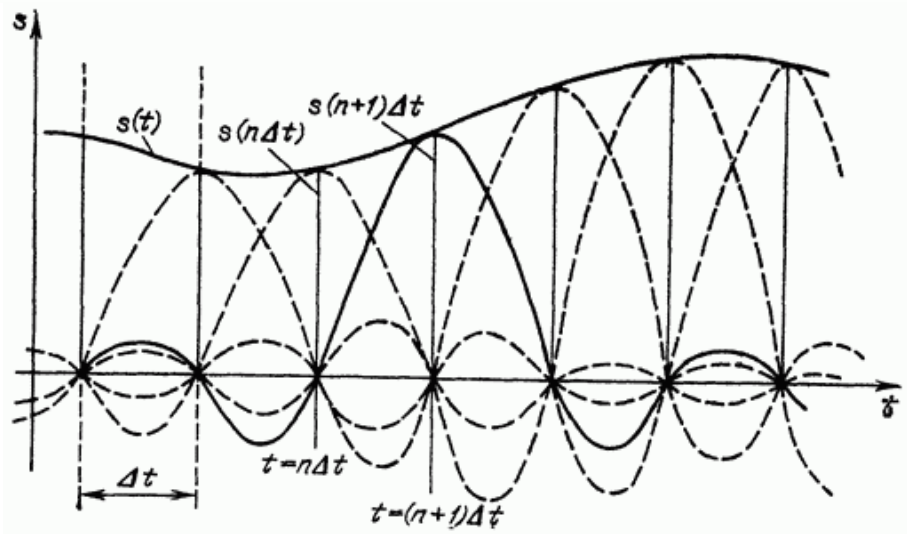
Data per one revolution (5400 sec) = 7569504 bits ≈ 1 Mb
@ 1200 bit/s, 420 sec window ≈ 15 passes

NYQUIST–SHANNON SAMPLING THEOREM

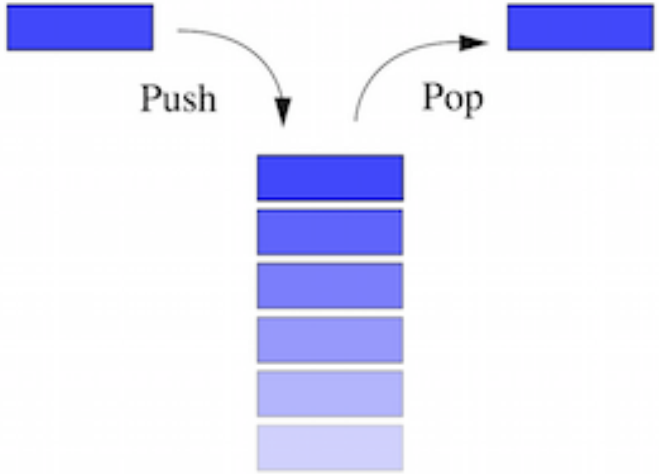
$$s(t) = \sum_{n=-\infty}^{+\infty} s\left(\frac{n}{2F}\right) sinc\left(2\pi F\left(t - \frac{n}{2F}\right)\right), \qquad sinc(x) = \frac{sin(x)}{x}$$



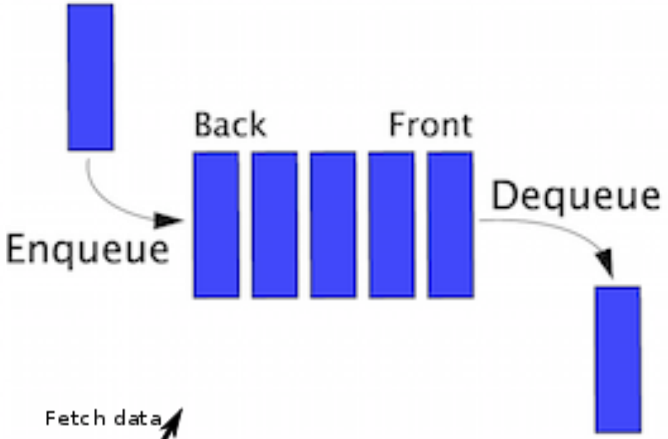


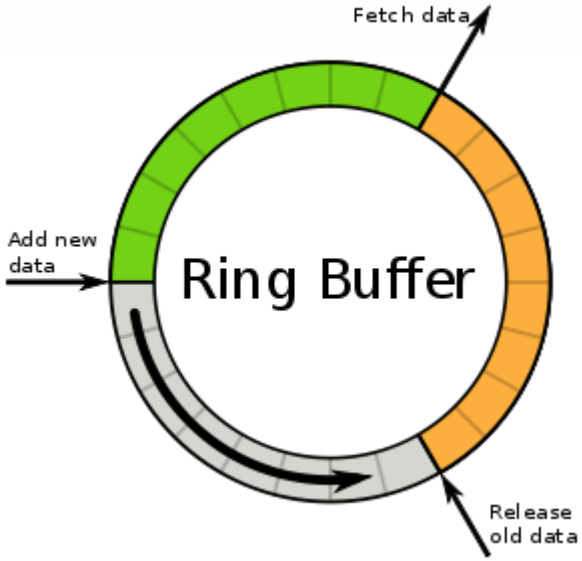


Stack

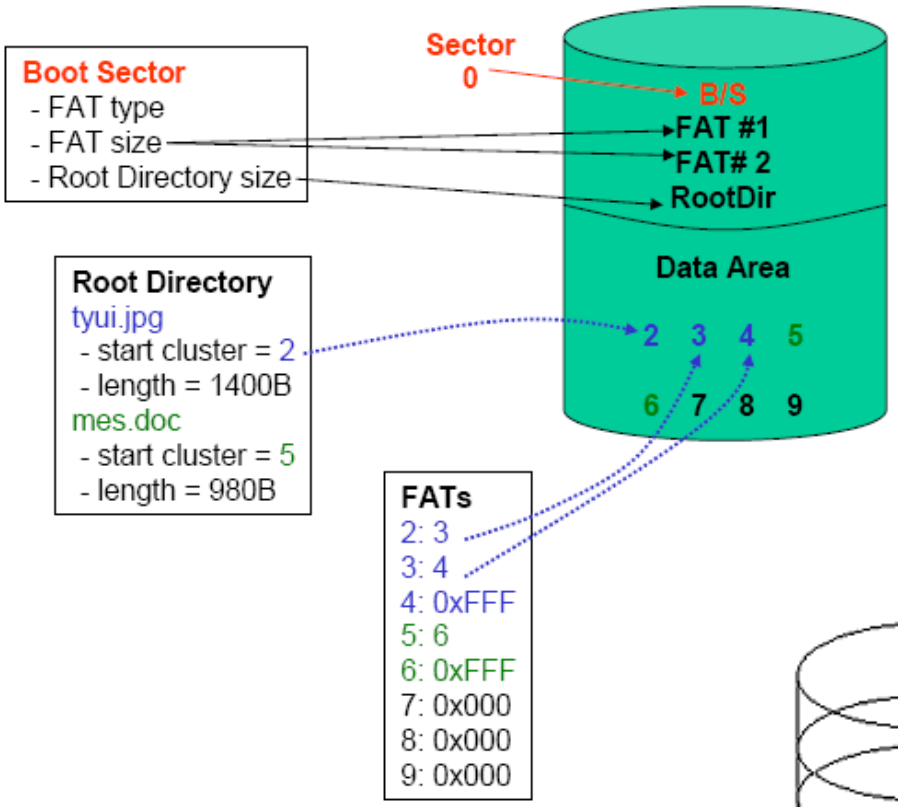

Push
Pop

Queue


Back
Front
Enqueue
Dequeue

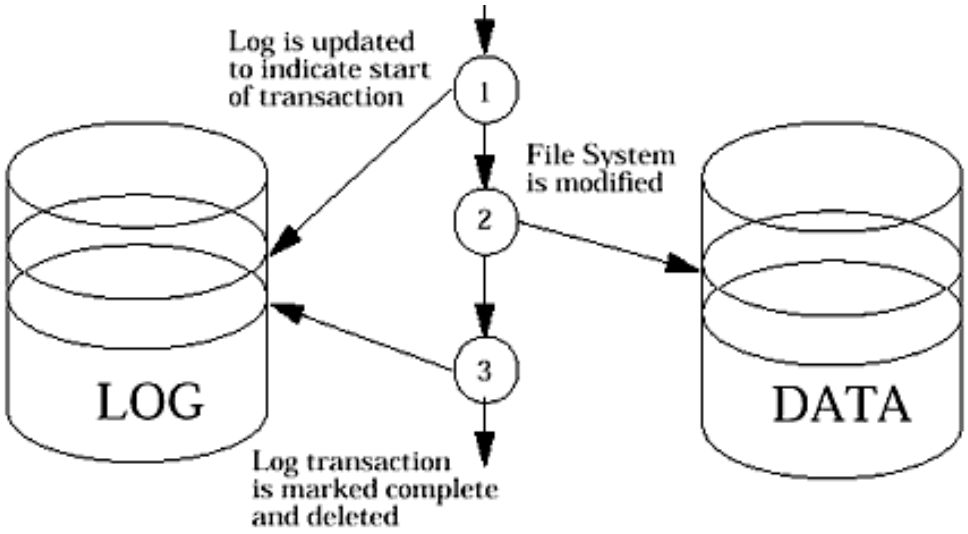

Fetch data
Add new data
Ring Buffer
Release old data

File Allocation Table (FAT)

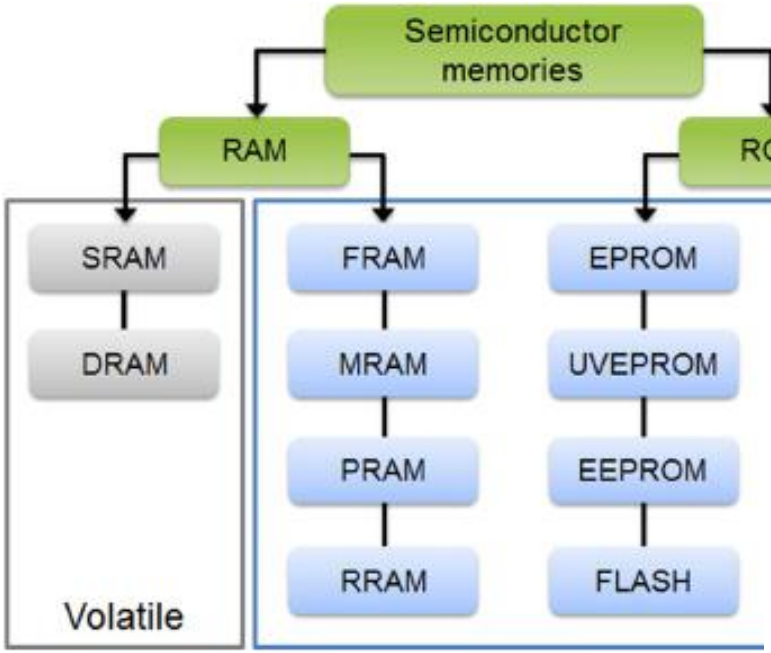


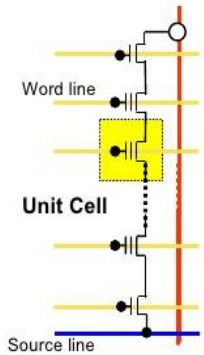
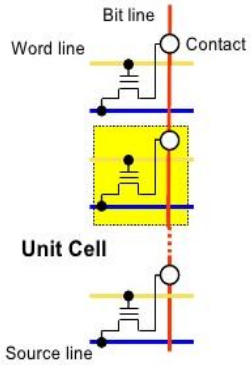
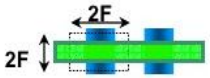
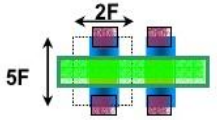
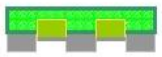
Application
File System Module
Low level disk I/O layer
Media

Journaling File System (JFS)



STORAGE TYPES



	NAND	NOR
Cell Array		
Layout		
Cross-section		
Cell size	$4F^2$	$10F^2$

COMMAND STRUCTURE

Request:	Header		Payload	CRC
	Packet ID	Command	Data	

Response:	Header		Payload	CRC
	Packet ID	Response ID	Data	

- Reboot
- Get status
- Get *N* bytes with *M* step from file *F*
- Get *N* bytes starts from byte *S* from file *F* with file step *M*
- Delete file
- Free memory space
- Format flash drive
- I2C direct read/write
- OBC control and status

Cortex Microcontroller Software Interface Standard

Aims:

- improving code reusage (portability) on other Cortex microcontrollers

- offers the possibility of using standard code by third party software developing firms

- shorten time of code development

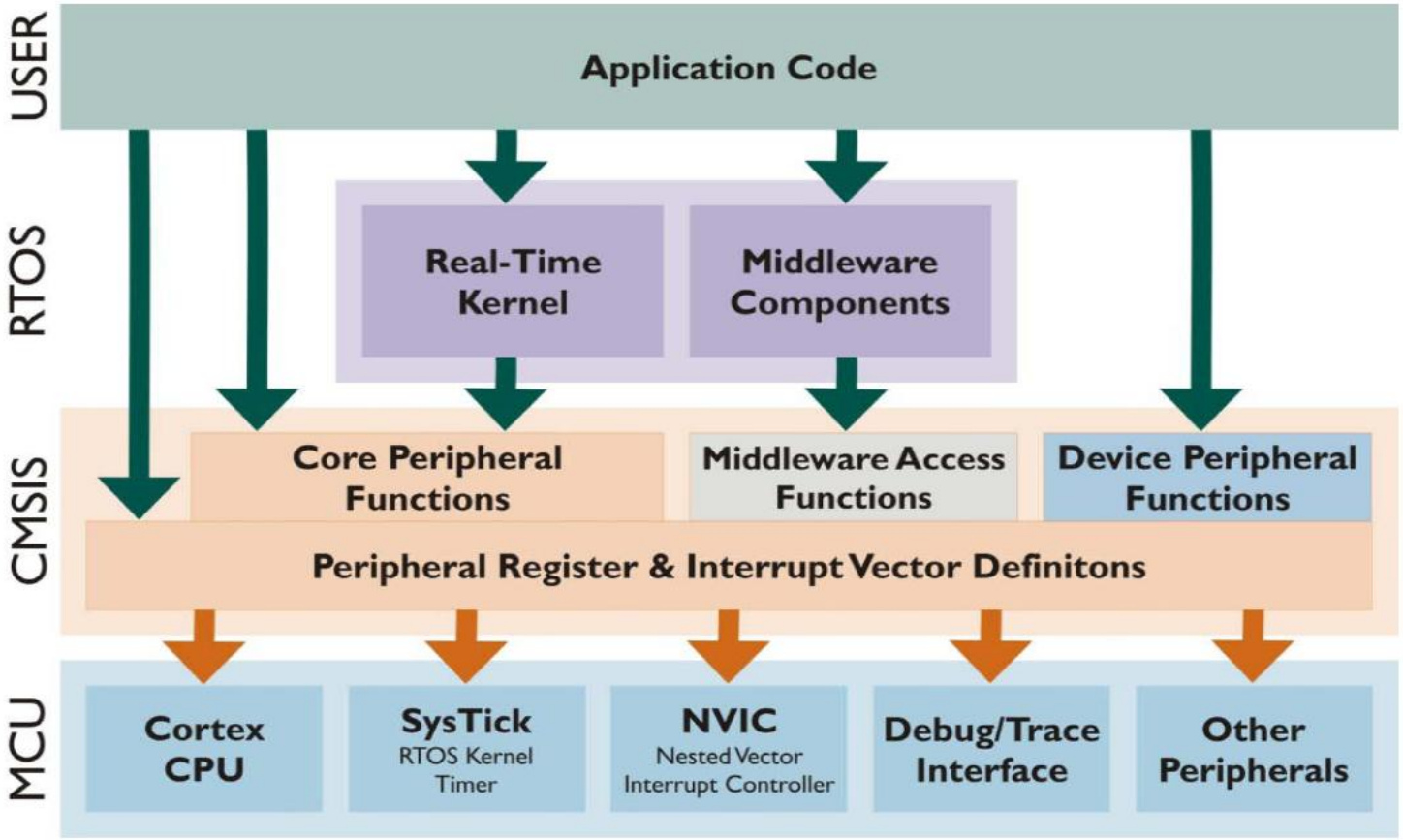
- using of different compilers

- Compatibility of code obtained from different sources

Cortex Microcontroller Software Interface Standard

- unified registers names (NVIC, SYSTICK, MPU)
- unified exception names
- unified structure of header files
- unified system initialization process (SystemInit();)
- standard intstrinc functions
- common functions for data exchange (UART, SPI, Ethernet)
- unified method of system clock frequency determination (global variable SystemFrequency)

CMSIS - Structure



CMSIS – Files for Peripheral Access Layer

Compiler Vendor-Independent Files:

- **Cortex-Mx Core Files (provided by ARM)**
  - core_cm3.h+core_cm3.c core_cm0.h+core_cm0.c
- **Device-specific Files (provided by Silicon Vendors)**
  - Register Header File (*device*.h)
  - System Startup File (system_*device*.c)
- **Compatible with all supported Compilers (IAR, RealView, GNU, …)**

Compiler-Vendor + Device-Specific Startup File:

- **Device Specific Compiler Startup Code (provided by Silicon Vendors)**
  - startup_*device*.s

CMSIS Files are available via www.onARM.com:

- **Device Database that lists all available devices**
  - CMSIS Files can be downloaded

CMSIS – Example

```
#include <device.h>                          // file name depends on device

void SysTick_Handler (void) {                // SysTick Interrupt Handler
   ;
}


void TIM1_UP_IRQHandler (void) {             // Timer Interrupt Handler
  ;
}

void timer1_init(int frequency) {            // set up Timer (device specific)
  NVIC_SetPriority (TIM1_UP_STM_IRQn, 1); // Set Timer priority
  NVIC_EnableIRQ (TIM1_UP_STM_IRQn);         // Enable Timer Interrupt
}

void main (void) {
  SystemInit ();                             // global system setup

  if (SysTick_Config (SystemFrequency / 1000)) { // SysTick 1mSec
       : // Handle Error
  }
  timer1_init ();                            // setup device specific timer
}
```

САМАРСКИЙ УНИВЕРСИТЕТ
SAMARA UNIVERSITY

THANK YOU

34, Moskovskoye shosse, Samara, 443086, Russia
Tel.: +7 (846) 335-18-26, fax: +7 (846) 335-18-36
www.ssau.ru, e-mail: ssau@ssau.ru