Forum Internacional de Software Livre v12

Hardware for embedded Linux devices

Daniel Bristot de Oliveira
daniel@bristot.eti.br
bristot@OFTC / #linux-rt

Who I'm and what we will talk about?

Why companies choose Linux for embedded systems?

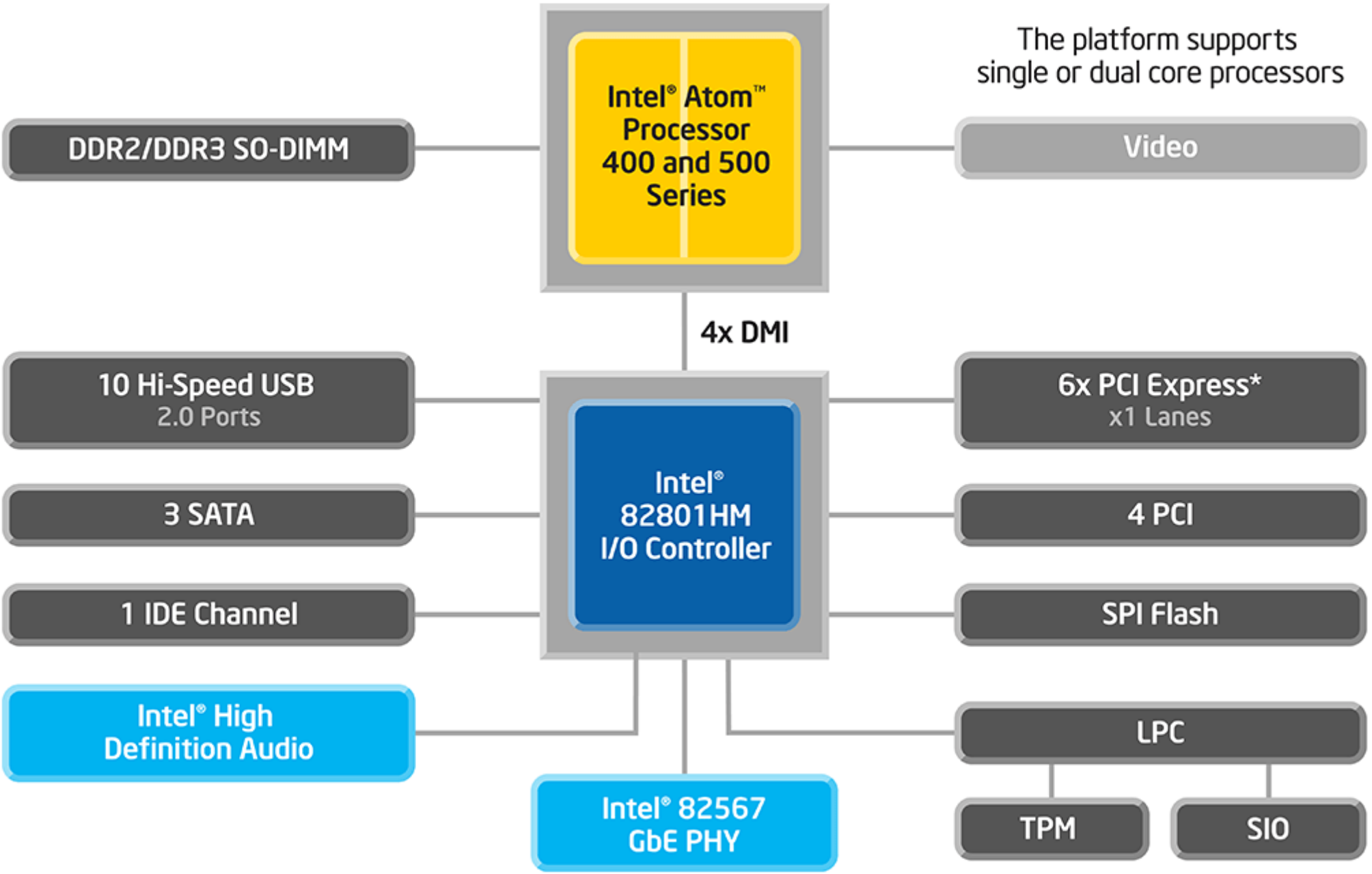
Many Users and developers, quality, available applications, open source, vendor independency, development tools and environment, stable ABI...

Hardware support!

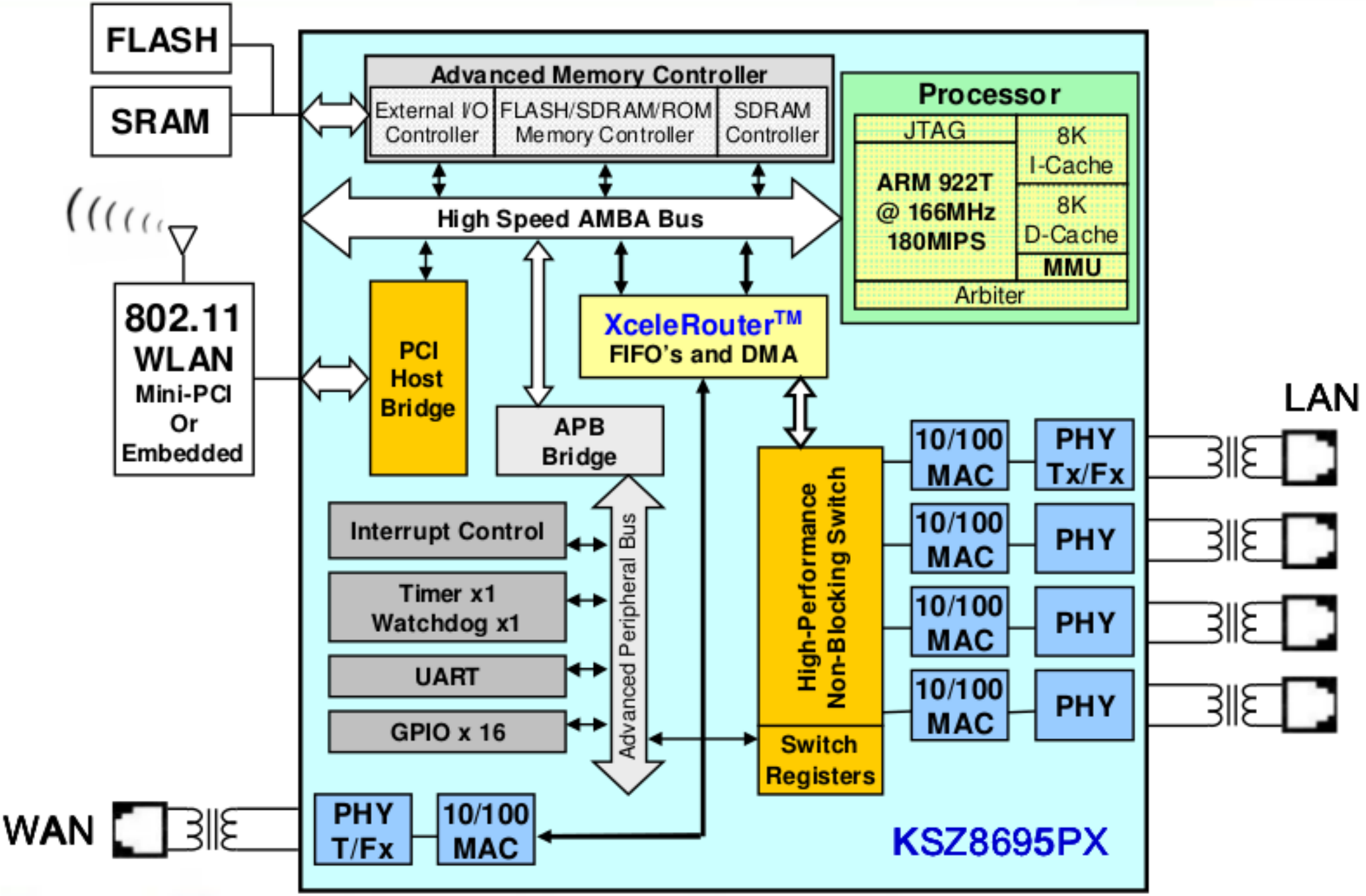
What classify a system as a **Embedded** system?

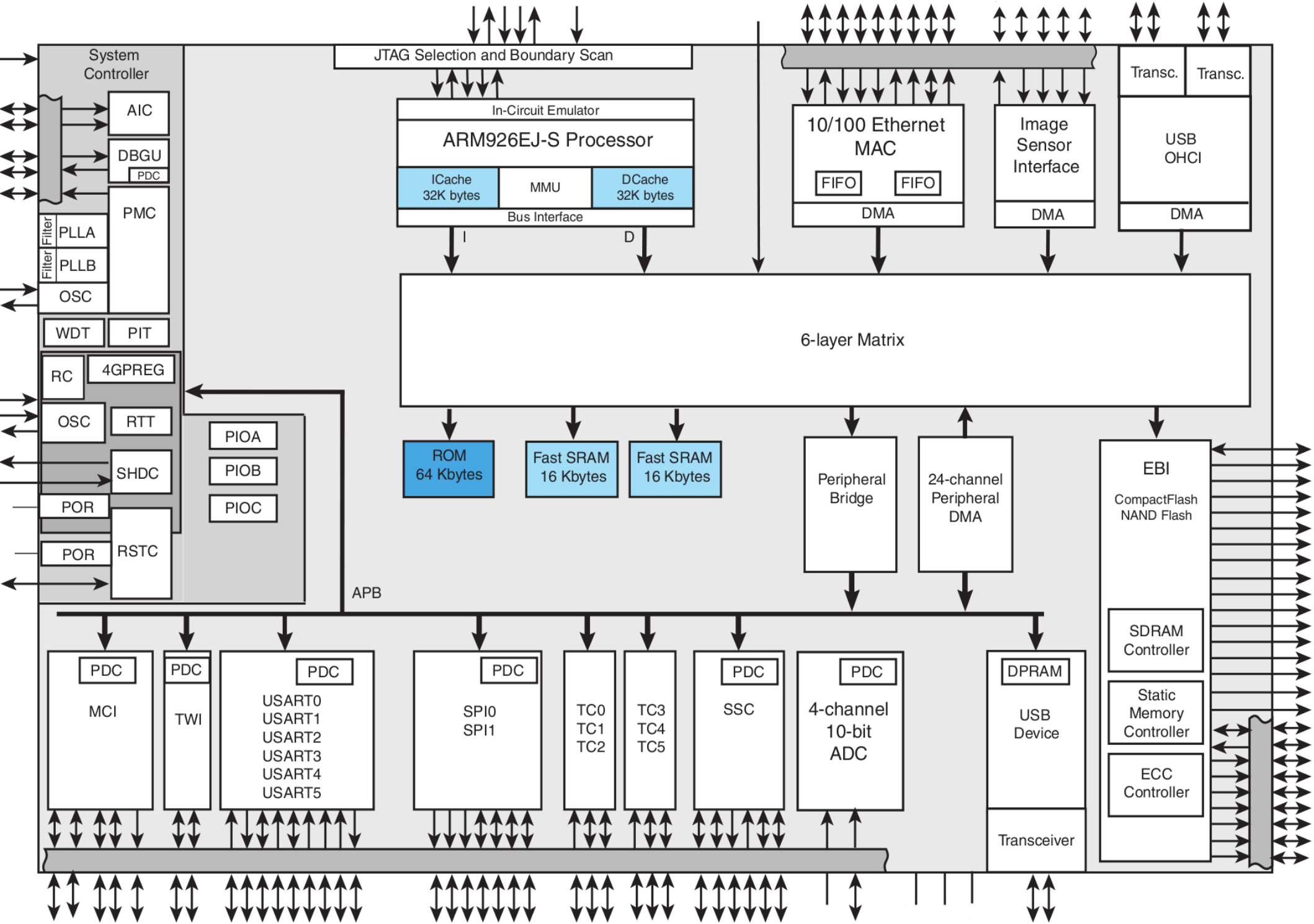
A counterexample:

General Purpose System


The platform supports
single or dual core processors
Intel® Atom™
Processor
400 and 500
Series
DDR2/DDR3 SO-DIMM
Video
4x DMI
10 Hi-Speed USB
2.0 Ports
6x PCI Express*
x1 Lanes
Intel®
82801HM
I/O Controller
3 SATA
4 PCI
1 IDE Channel
SPI Flash
Intel® High
Definition Audio
LPC
Intel® 82567
GbE PHY
TPM
SIO

A Embedded System example


FLASH
SRAM
802.11
WLAN
Mini-PCI
Or
Embedded
Advanced Memory Controller
External I/O Controller
FLASH/SDRAM/ROM Memory Controller
SDRAM Controller
Processor
JTAG
ARM 922T
@ 166MHz
180MIPS
8K
I-Cache
8K
D-Cache
MMU
Arbiter
High Speed AMBA Bus
PCI
Host
Bridge
XceleRouter™
FIFO's and DMA
APB
Bridge
Interrupt Control
Timer x1
Watchdog x1
UART
GPIO x 16
Advanced Peripheral Bus
High-Performance
Non-Blocking Switch
Switch
Registers
10/100
MAC
PHY
Tx/Fx
PHY
LAN
WAN
PHY
T/Fx
KSZ8695PX


System Controller
AIC
DBGU
PDC
PMC
Filter
PLLA
Filter
PLLB
OSC
WDT
PIT
RC
4GPREG
OSC
RTT
SHDC
POR
POR
RSTC
PIOA
PIOB
PIOC
JTAG Selection and Boundary Scan
In-Circuit Emulator
ARM926EJ-S Processor
ICache 32K bytes
MMU
DCache 32K bytes
Bus Interface
I
D
10/100 Ethernet MAC
FIFO
FIFO
DMA
Image Sensor Interface
DMA
Transc.
Transc.
USB OHCI
DMA
6-layer Matrix
ROM 64 Kbytes
Fast SRAM 16 Kbytes
Fast SRAM 16 Kbytes
Peripheral Bridge
24-channel Peripheral DMA
EBI
CompactFlash
NAND Flash
SDRAM Controller
Static Memory Controller
ECC Controller
APB
PDC
MCI
PDC
TWI
PDC
USART0
USART1
USART2
USART3
USART4
USART5
PDC
SPI0
SPI1
TC0
TC1
TC2
TC3
TC4
TC5
PDC
SSC
PDC
4-channel 10-bit ADC
DPRAM
USB Device
Transceiver

What is System on Chip and why it make success on embedded world?

It generate a problem: there's a lot o options of SoC on the market! It make us life hard...(?!), what SoC and peripherals choose?

Embedded Architetures examples

ARM + NEON + Jazzele + DSP & SIMD extensions

MIPS

AD BlackFin

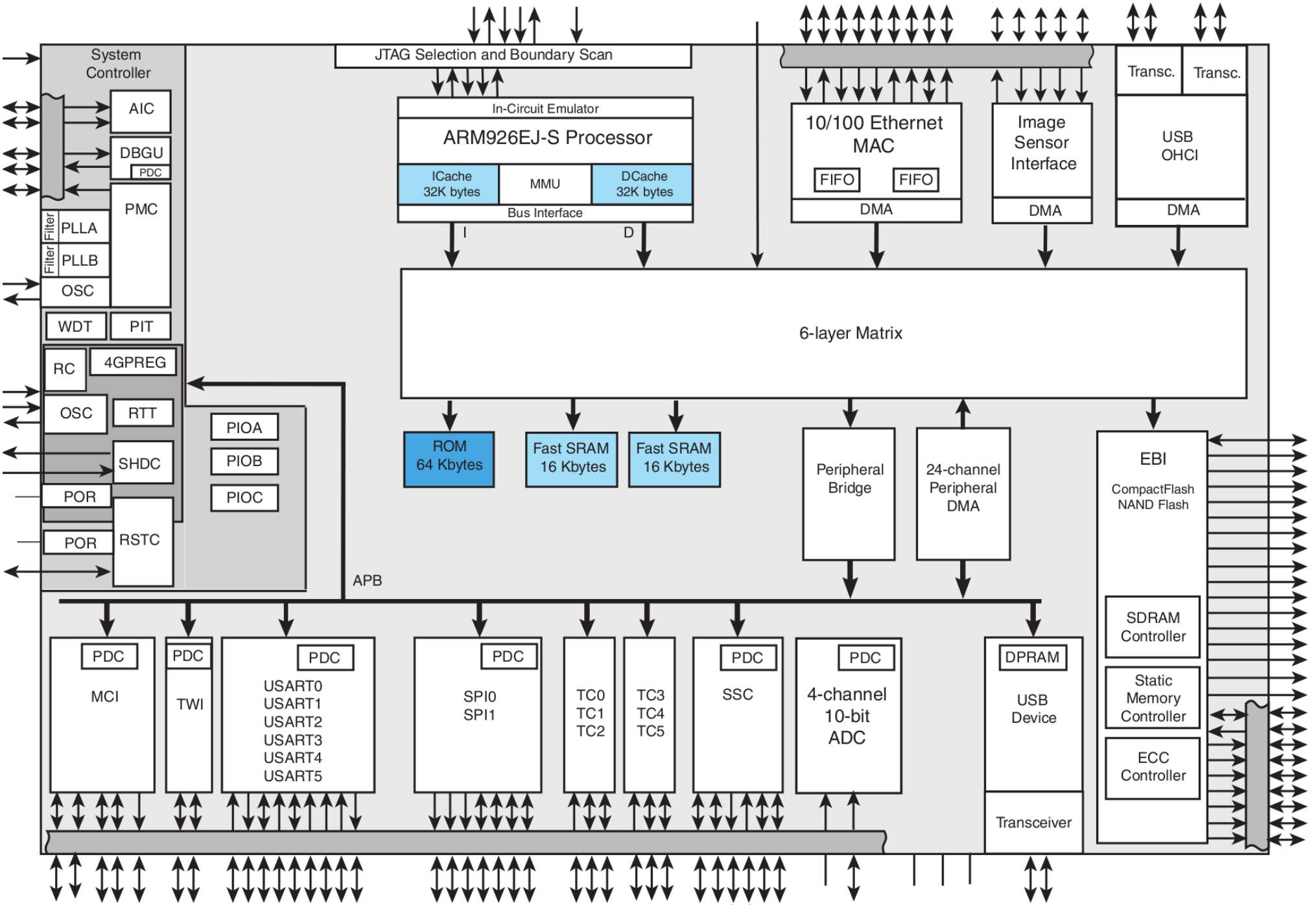
Microblaze

Intel ATOM (???)

The most important part of a SoC...

The **peripherals** (that many times is not on periphery)

Example System:

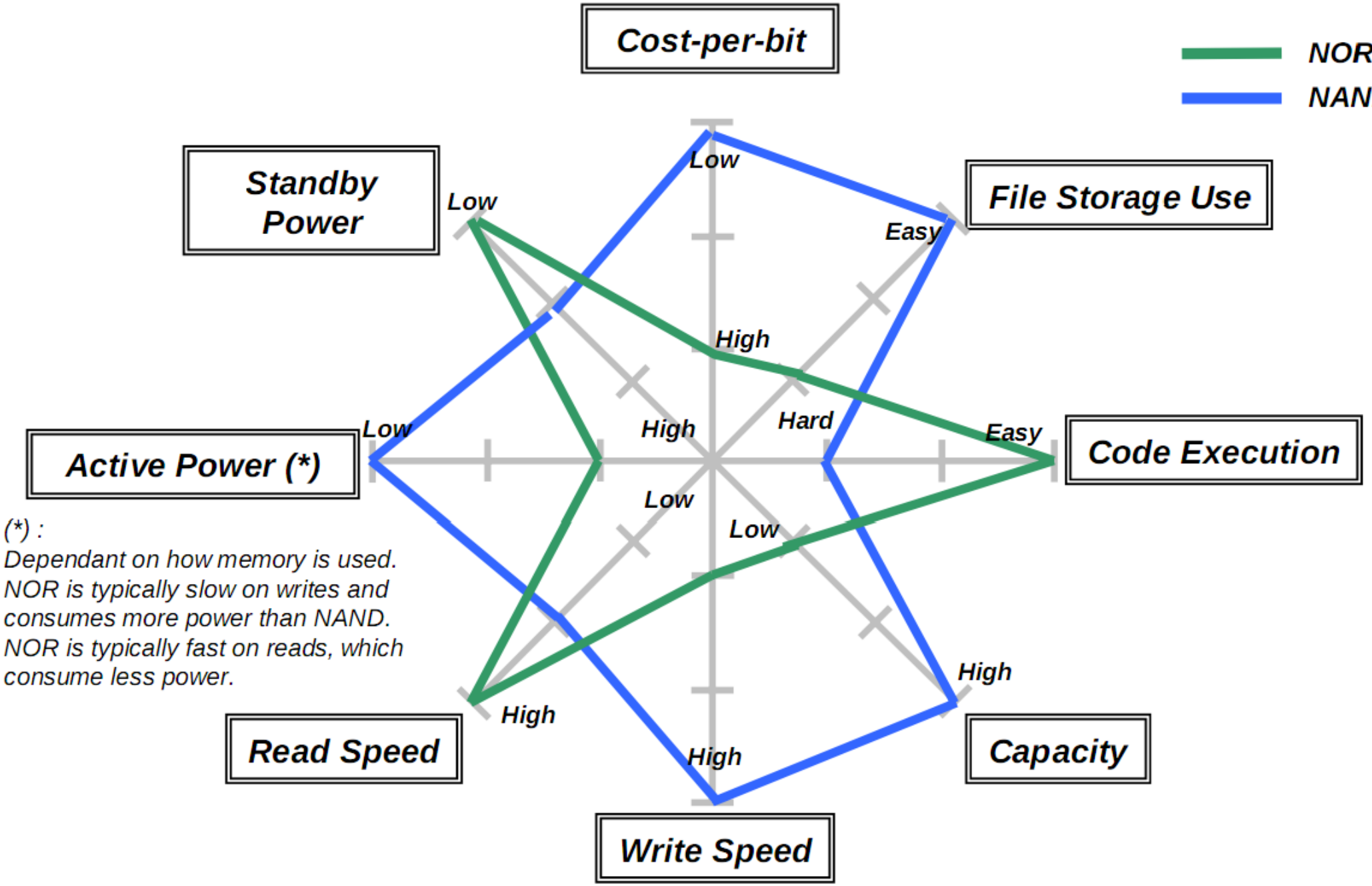


CPU, Cache and BUS Speed

Storage

NAND and NOR

Flash memories


Cost-per-bit
NOR
NAND
Standby Power
Low
File Storage Use
Low
Easy
High
Active Power (*)
Low
High
Hard
Easy
Code Execution
Low
Low
(*) :
Dependant on how memory is used.
NOR is typically slow on writes and
consumes more power than NAND.
NOR is typically fast on reads, which
consume less power.
High
Read Speed
High
Capacity
High
Write Speed

Serial Flash and SSD

Flash File Systems

System BOOT

We don't need the BIOS...

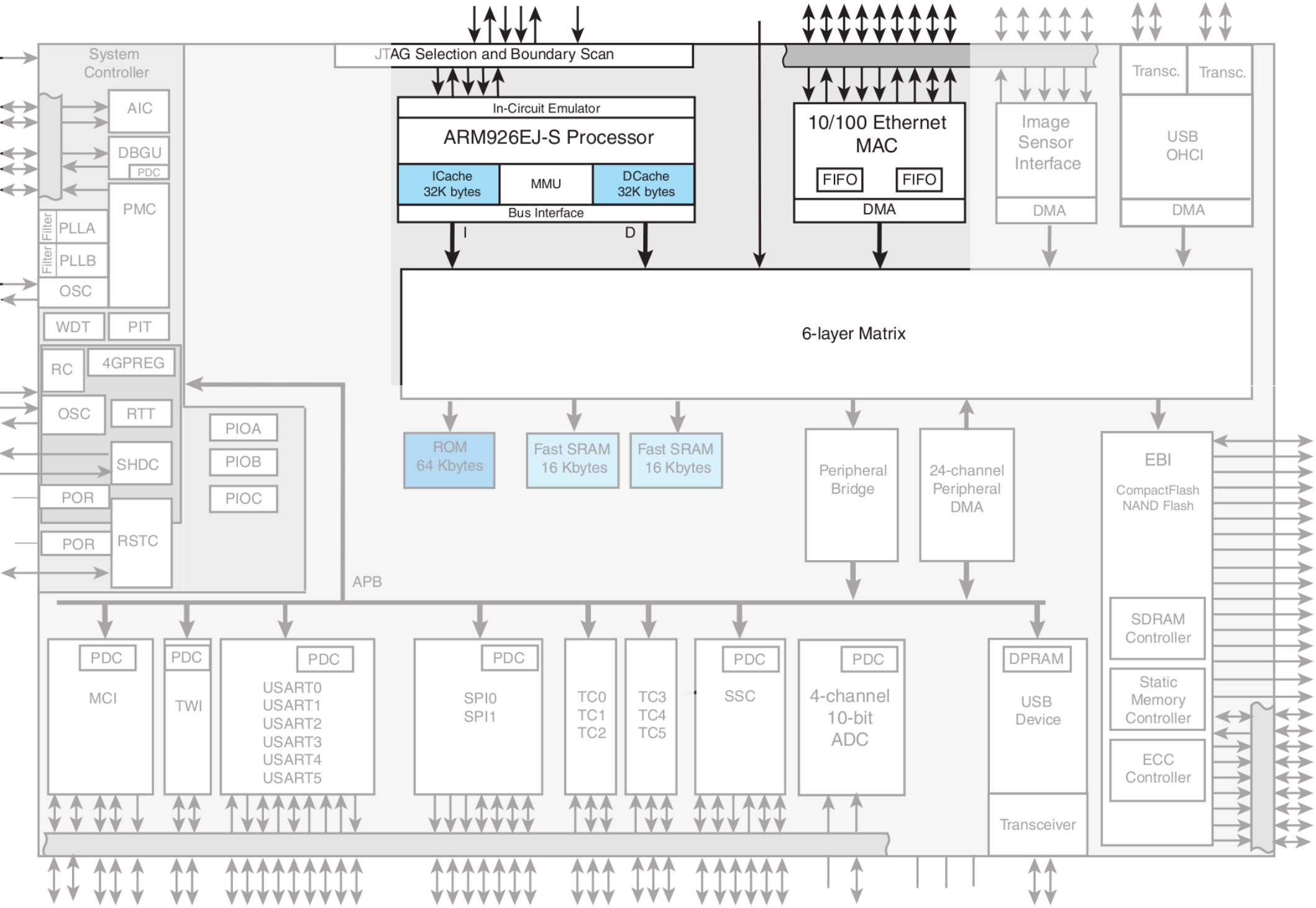
perhaps because the bios is more expensive that all the system :)

Bootstrap and Bootloader is all we need

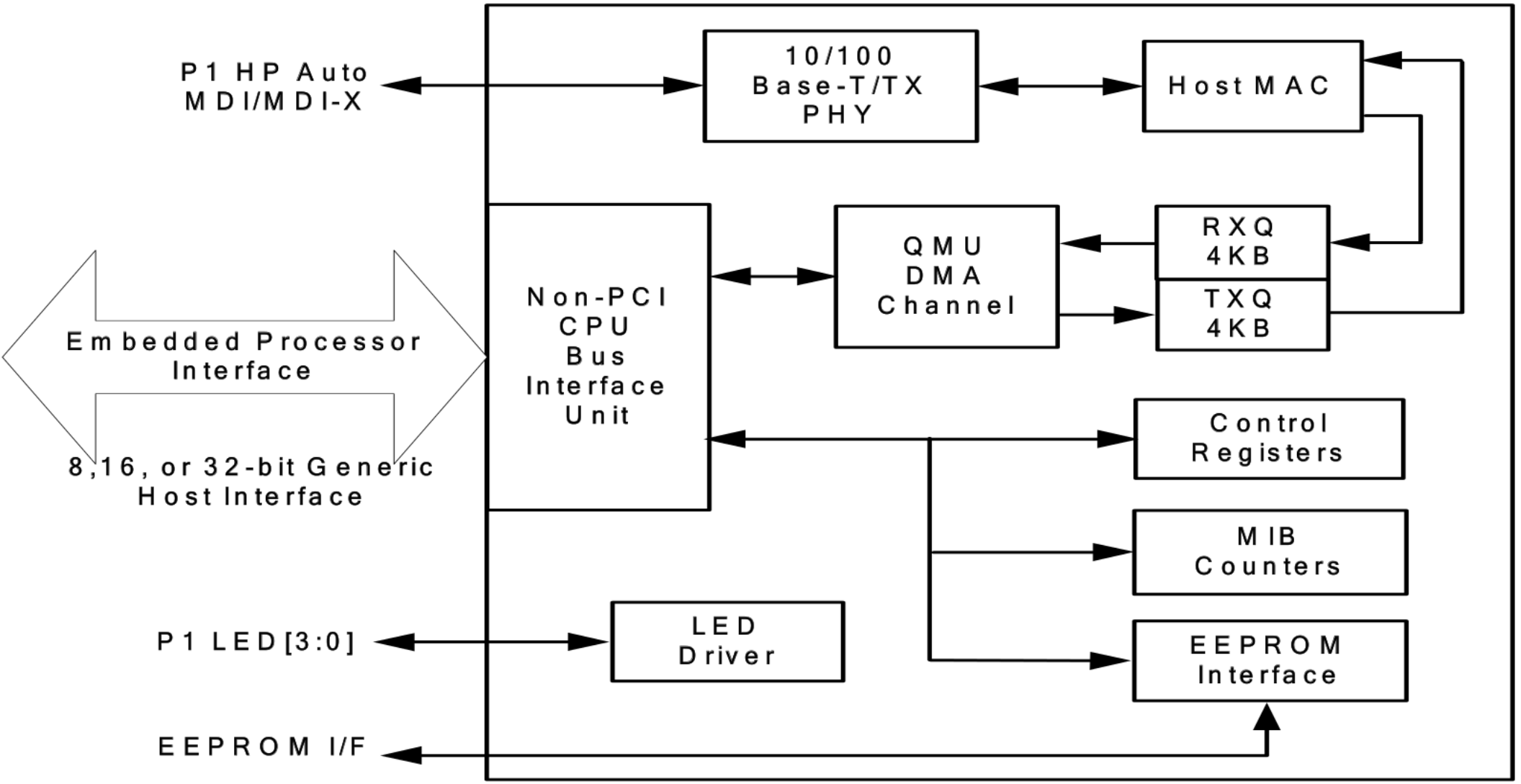
But the hardware need know US!

Pay attention with hardware boot strategies

Connector + PHY + MAC = Network Interface

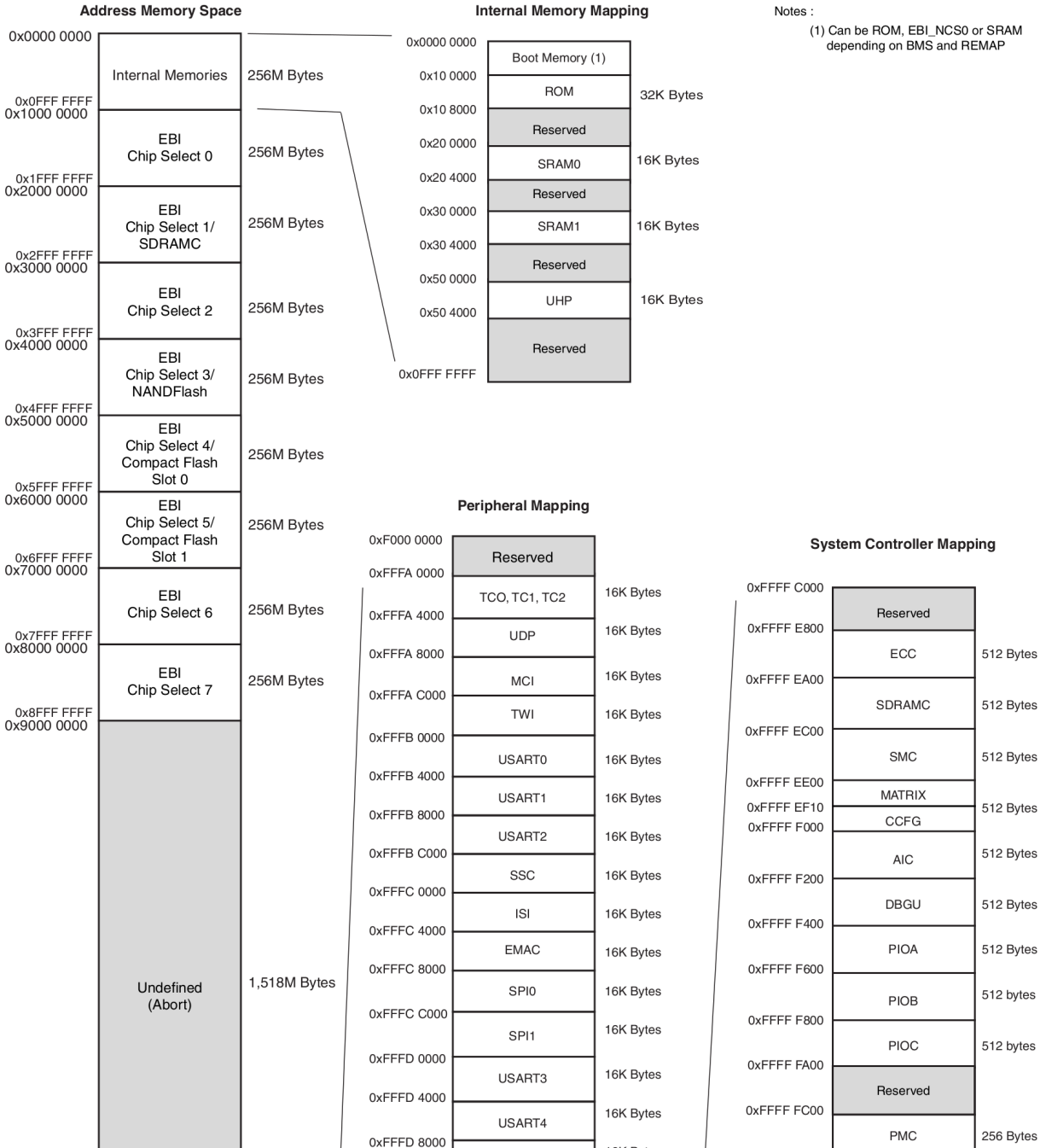

System Controller
AIC
DBGU
PDC
PMC
Filter
PLLA
Filter
PLLB
OSC
WDT
PIT
RC
4GPREG
OSC
RTT
SHDC
POR
POR
RSTC
PIOA
PIOB
PIOC
JTAG Selection and Boundary Scan
In-Circuit Emulator
ARM926EJ-S Processor
ICache 32K bytes
MMU
DCache 32K bytes
Bus Interface
I
D
10/100 Ethernet MAC
FIFO
FIFO
DMA
Image Sensor Interface
DMA
Transc.
Transc.
USB OHCI
DMA
6-layer Matrix
ROM 64 Kbytes
Fast SRAM 16 Kbytes
Fast SRAM 16 Kbytes
Peripheral Bridge
24-channel Peripheral DMA
EBI
CompactFlash
NAND Flash
SDRAM Controller
Static Memory Controller
ECC Controller
APB
PDC
MCI
PDC
TWI
PDC
USART0
USART1
USART2
USART3
USART4
USART5
PDC
SPI0
SPI1
TC0
TC1
TC2
TC3
TC4
TC5
PDC
SSC
PDC
4-channel 10-bit ADC
DPRAM
USB Device
Transceiver

External MAC + PHY



Advanced and fun MAC options :)

Memory Mapped devices



Memory Mapped Devices

Some considerations about DMA Controller

When and where use DMA?

DMA Controller scheduler

IRQ and Linux Kernel
(a little detail)

Aleatory tips space...

JTAG Debugger

Kernel options

Compiler

Rootfs...

Questions?

Thank you all!