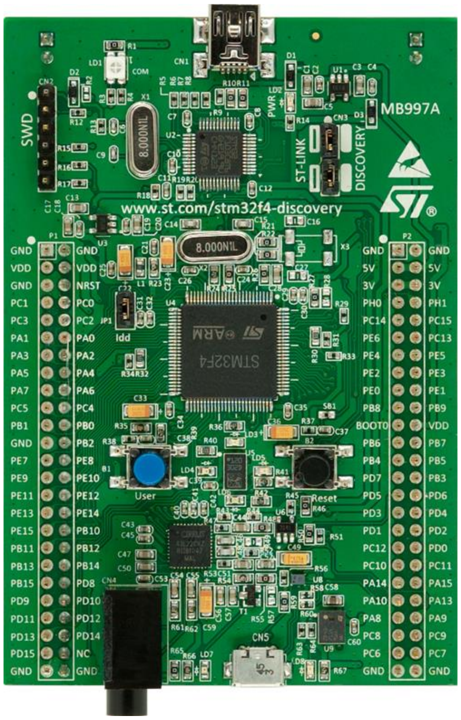
Chapter 4
INTRODUCTION

1. Microcontroller families:

- 8bit:
    - + 8051 : Intel, Atmel, Philips, Dallas, Analog Devices,…
    - + HC11 : Motorola
    - + AVR : Atmel
    - + PIC : Microchip
- 16bit:
    - + 80C196 : Intel
    - + 80C166 : Siemens
    - + PIC24 : Microchip
    - + F24xx : Texas Instruments
- 32bit:
    - + HC12 : Motorola
    - + ARM : Atmel, Philips, ST, Analog Devices, …
    - + PIC32 : Microchip
    - + F28xx : Texas Instruments

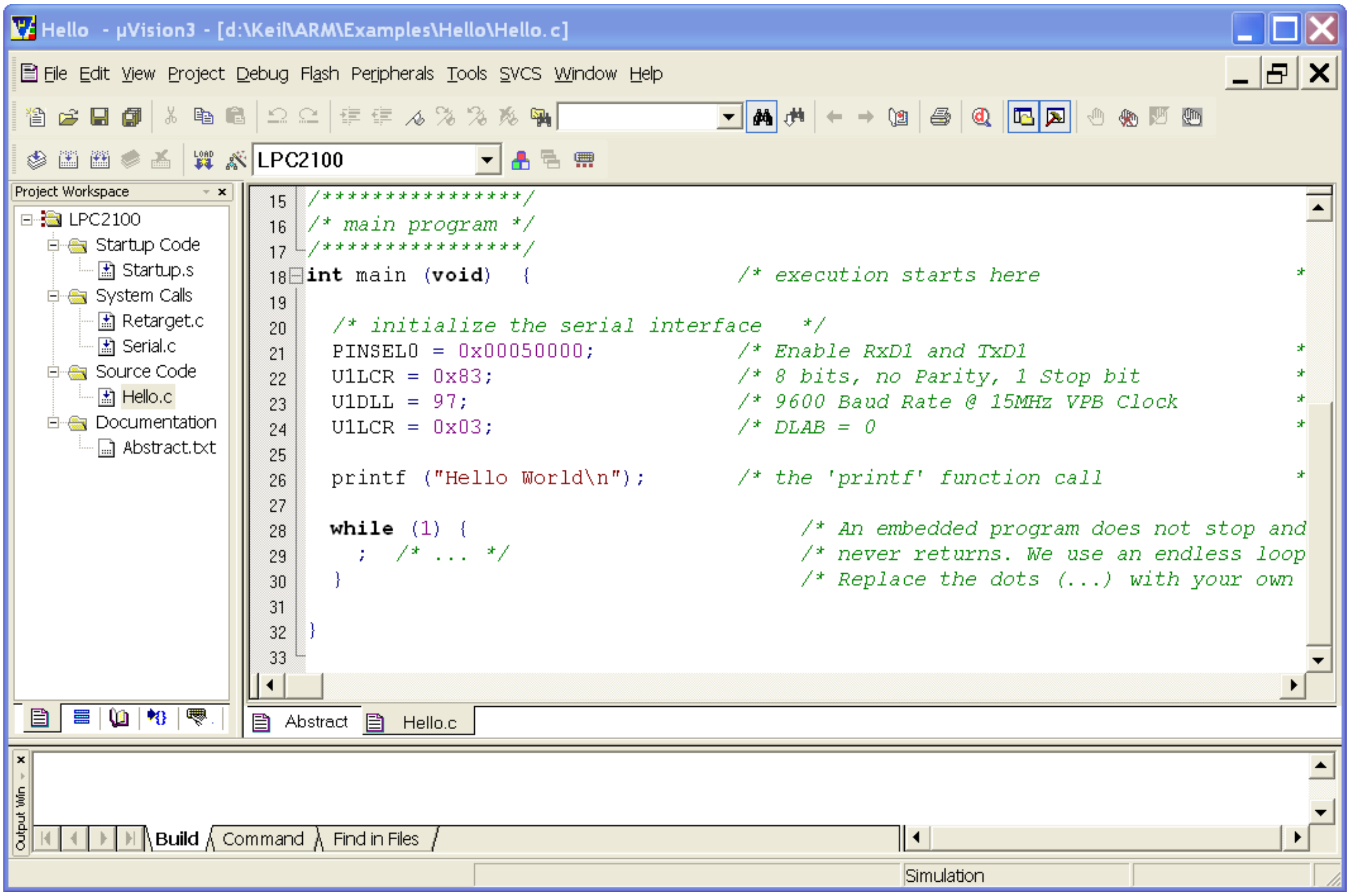
2. Outline

2.1. ARM Cortex-M4 (ST STM32F4xx) – STM32F4Discovery



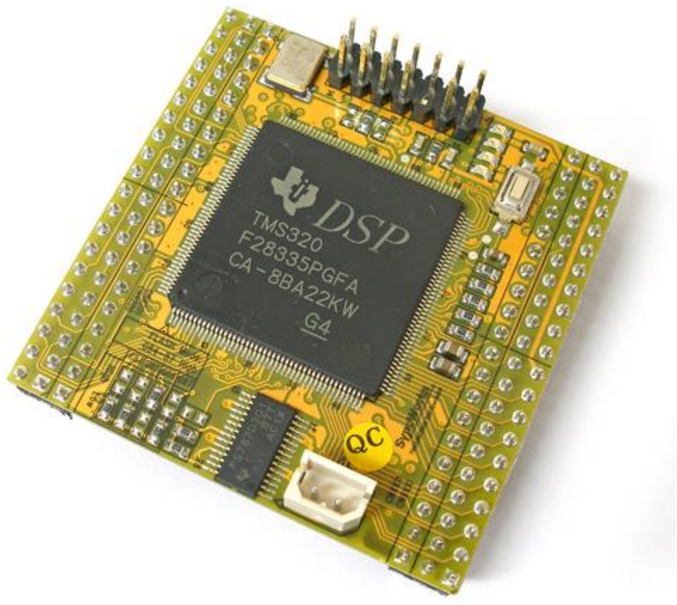
- 1MB Flash, 192KB SRAM
- ARM 32-bit Cortex™-M4F CPU with FPU, frequency up to 168 MHz
- Full Speed USB 2.0 Port
- Ethernet LAN 10/100Mb
- 3 Channels 12-bit ADC
- 2 Channels 12-bit DAC
- 2 Channels standard CAN network
- 4 SCI (UART), 3 SPI, 3 I2C
- 8- to 14-bit parallel camera interface up to 54 Mbytes /s
- 16 Channels DMA
- Single 3.3 V power supply

Compiler



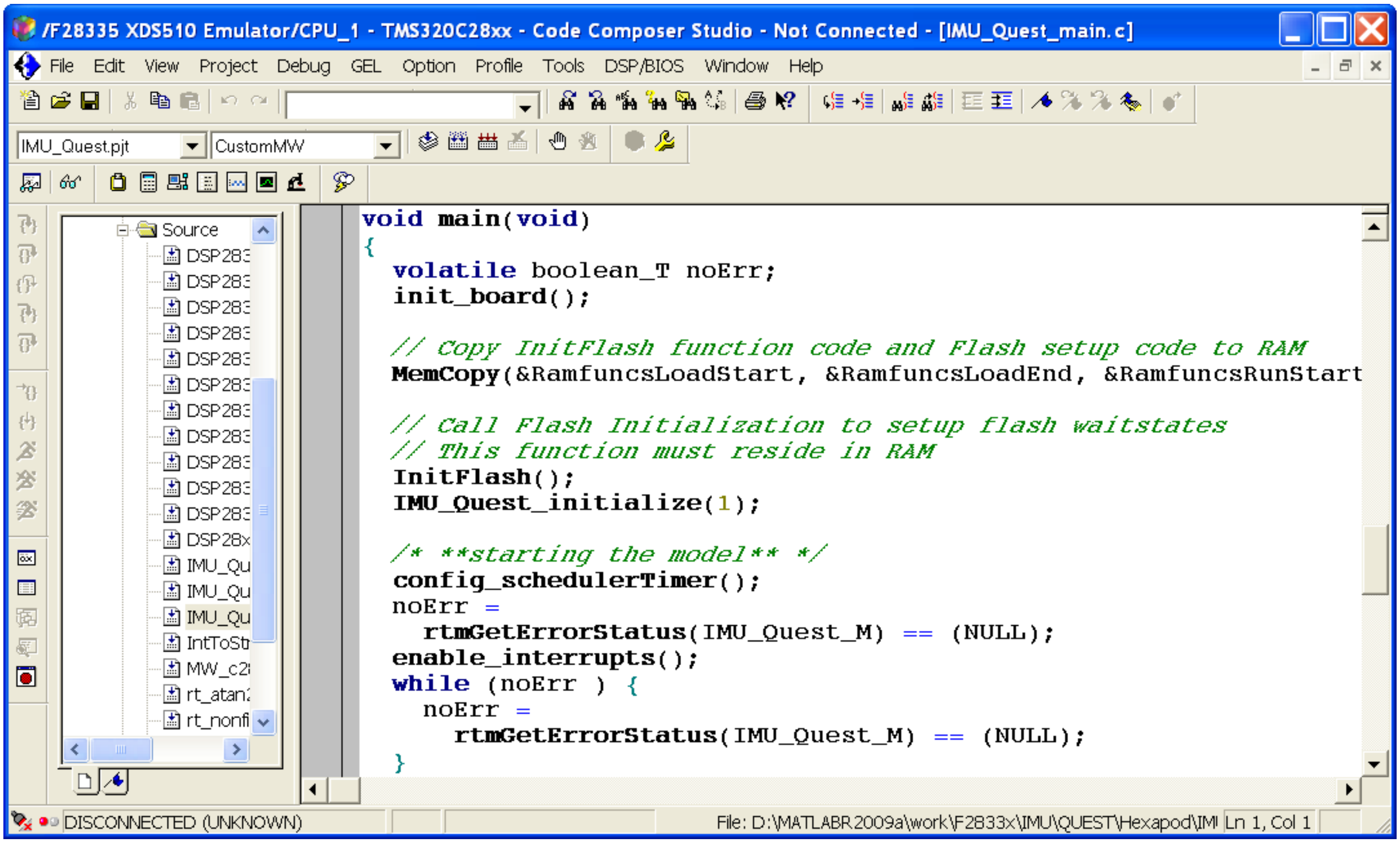
```
/****************/
/* main program */
/****************/
int main (void)  {                /* execution starts here                    *

  /* initialize the serial interface   */
  PINSEL0 = 0x00050000;           /* Enable RxD1 and TxD1                     *
  U1LCR = 0x83;                   /* 8 bits, no Parity, 1 Stop bit            *
  U1DLL = 97;                     /* 9600 Baud Rate @ 15MHz VPB Clock         *
  U1LCR = 0x03;                   /* DLAB = 0                                 *

  printf ("Hello World\n");       /* the 'printf' function call               *

  while (1) {                          /* An embedded program does not stop and
    ;  /* ... */                       /* never returns. We use an endless loop
  }                                    /* Replace the dots (...) with your own

}
```

2.2. DSP (Texas Instruments TMS320F28335)



- 512KB Flash
- 68KB SRAM
- Operating Speed up to 150 MHz
- Single-Precision Floating-Point
- 16 Channels 12-bit A/D
- 18 Channels PWM, 2 Encoder
- 2 Channels standard CAN network
- 3 SCI (UART), 1 SPI, 1 I2C
- Up to 88 IO pins
- 6 Channels DMA
- 1.9-V Core, 3.3-V I/O Design

Compiler



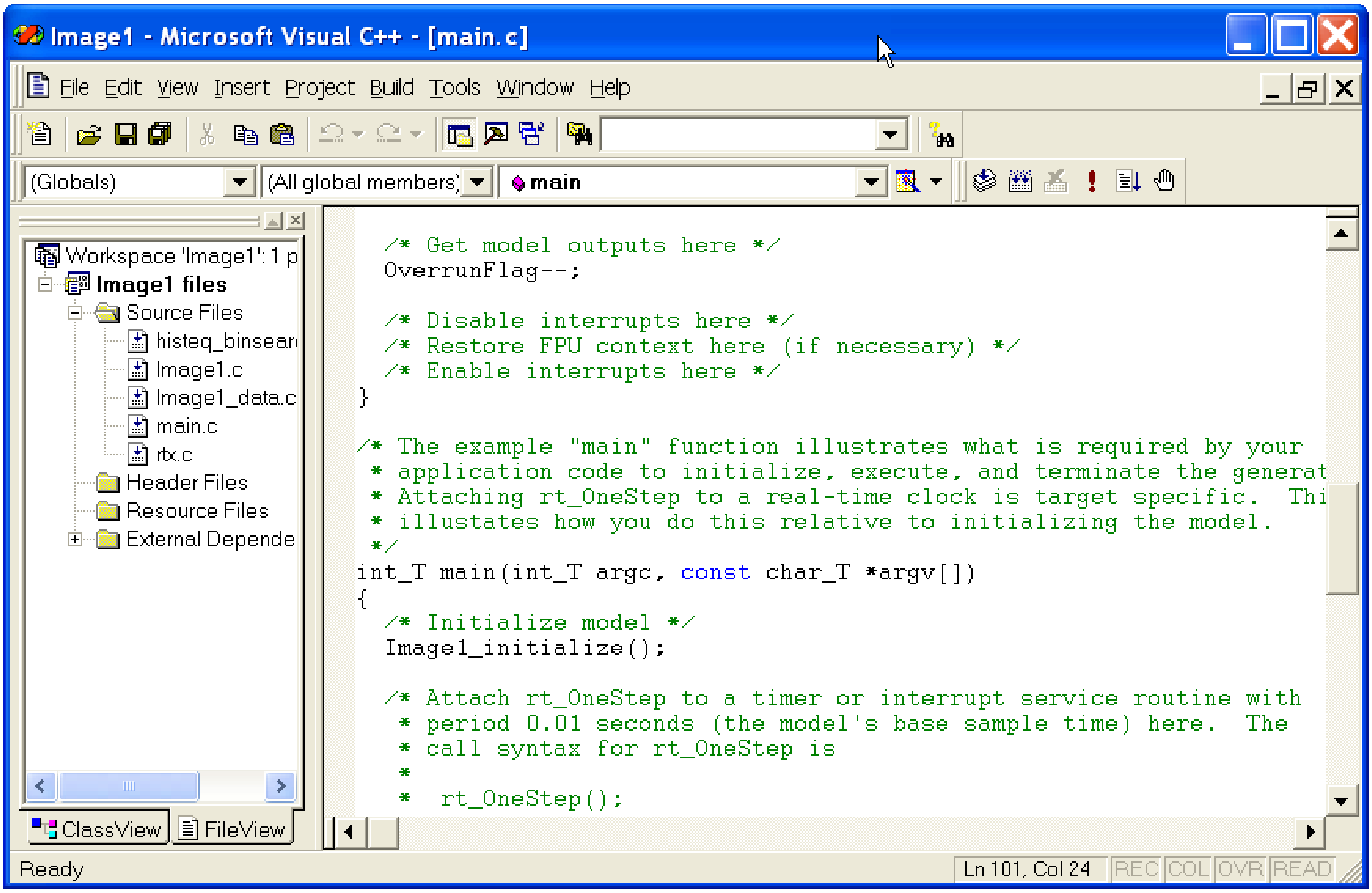
```
void main(void)
{
  volatile boolean_T noErr;
  init_board();

  // Copy InitFlash function code and Flash setup code to RAM
  MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd, &RamfuncsRunStart
  
  // Call Flash Initialization to setup flash waitstates
  // This function must reside in RAM
  InitFlash();
  IMU_Quest_initialize(1);

  /* **starting the model** */
  config_schedulerTimer();
  noErr =
    rtmGetErrorStatus(IMU_Quest_M) == (NULL);
  enable_interrupts();
  while (noErr ) {
    noErr =
      rtmGetErrorStatus(IMU_Quest_M) == (NULL);
  }
```

2.3. PC104 (Advantech PCM-3353)

- AMD Geode® LX800, 500 MHz
- Up to 2GB DDR 333MHz
- 4 serial RS-232 ports
- 1 parallel port, SPP/EPP/ECP mode
- Supports AC97 Audio stereo sound
- 4 USB 2.0
- 1 50-pin socket for CF Card type I
- 5-V, 12-V power supply

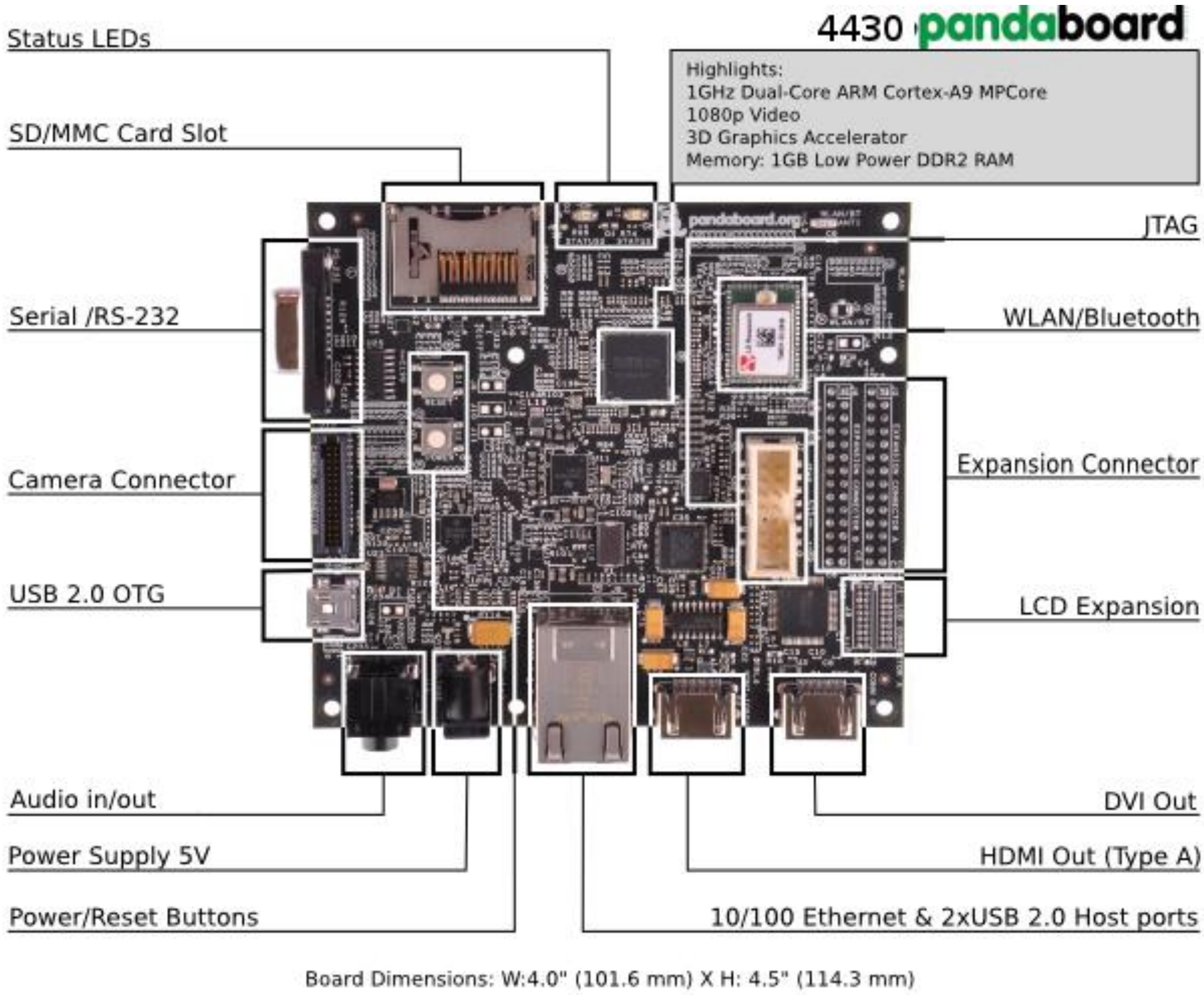
Compiler

DOS : Watcom C/C++
WIN : Visual C/C++



```
  /* Get model outputs here */
  OverrunFlag--;

  /* Disable interrupts here */
  /* Restore FPU context here (if necessary) */
  /* Enable interrupts here */
}

/* The example "main" function illustrates what is required by your
 * application code to initialize, execute, and terminate the generat
 * Attaching rt_OneStep to a real-time clock is target specific.  Thi
 * illustates how you do this relative to initializing the model.
 */
int_T main(int_T argc, const char_T *argv[])
{
  /* Initialize model */
  Image1_initialize();

  /* Attach rt_OneStep to a timer or interrupt service routine with
   * period 0.01 seconds (the model's base sample time) here.  The
   * call syntax for rt_OneStep is
   *
   *  rt_OneStep();
```

2.4. Beagle board, Panda board – NVIDIA chip Jetson Nano, TX2, Xavier



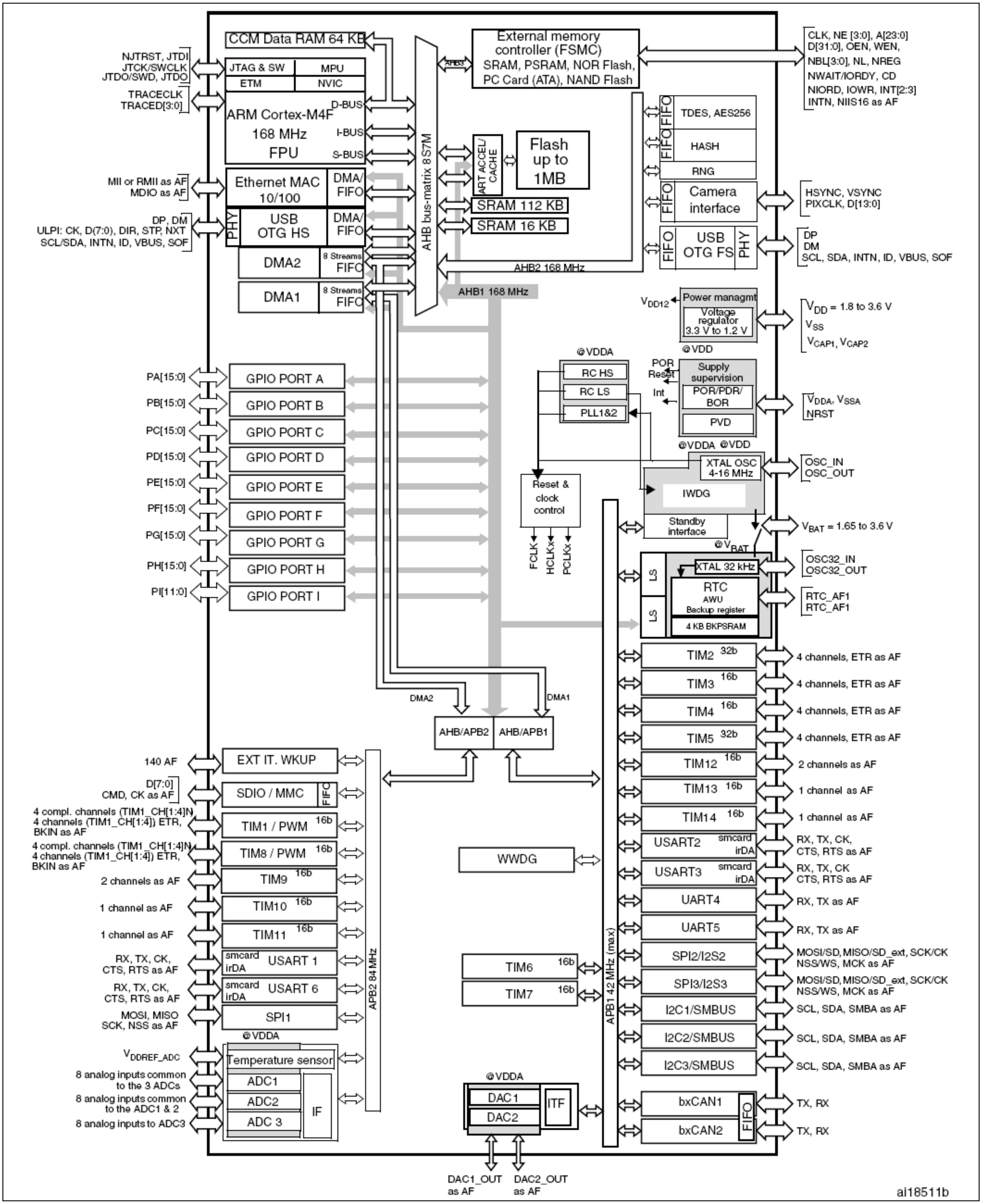
Compiler

Linux: Qtopia
Android:….

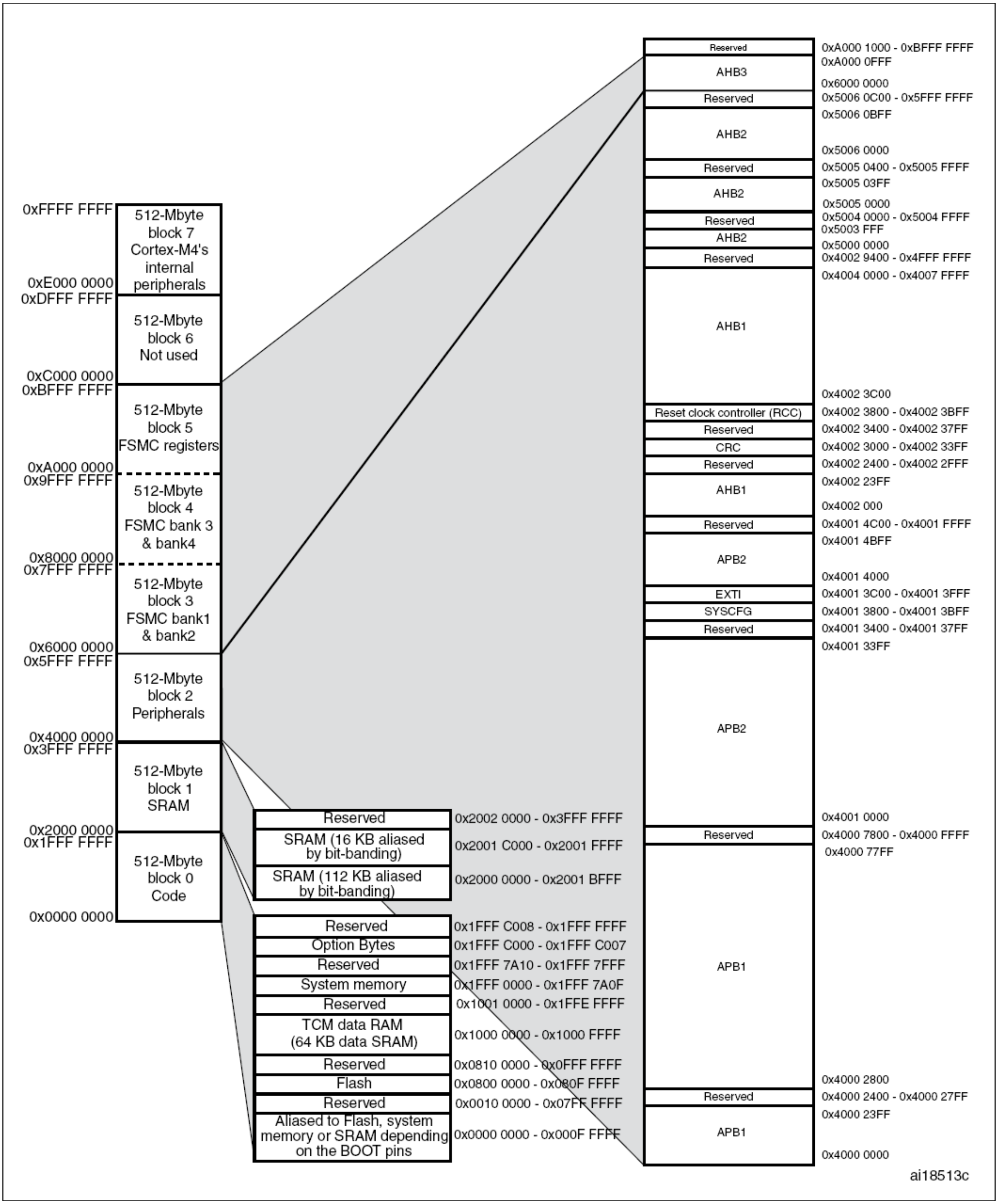
Chapter 2
ST ARM STM32F4

1. Features:

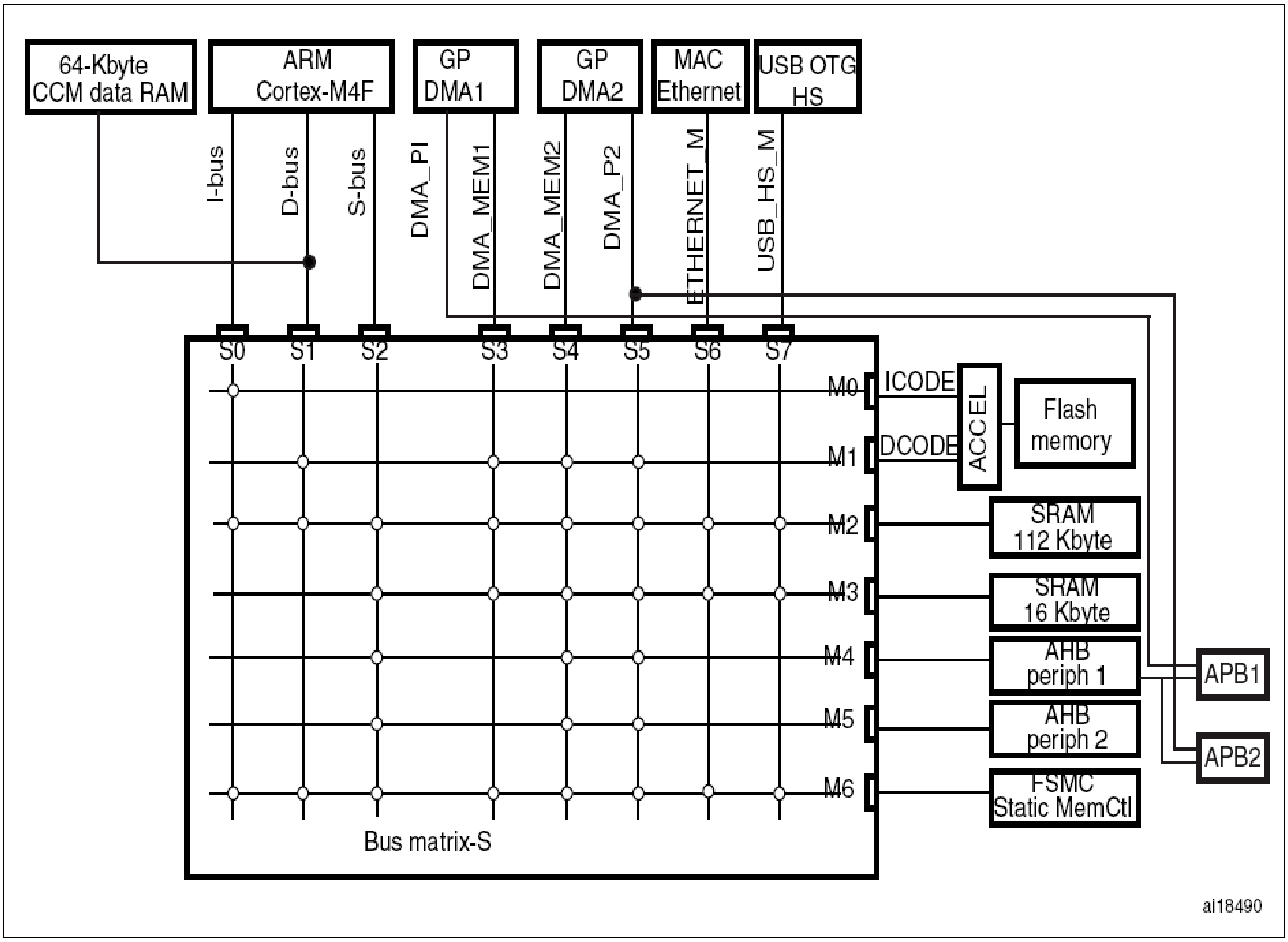
- Core: ARM 32-bit Cortex™-M4F CPU with FPU, Adaptive real-time accelerator (ART Accelerator™) allowing 0-wait state execution from Flash memory, frequency up to 168 MHz,
- Memories:
  - Up to 1 Mbyte of Flash memory.
  - Up to 192+4 Kbytes of
  - Flexible static memory controller supporting Compact Flash, SRAM,…
- 3×12-bit, 2.4 MSPS A/D converters: up to 24 channels and 7.2 MSPS
- 2×12-bit D/A converters
- General-purpose DMA: 16-stream DMA controller with FIFOs and burst support
- Up to 17 timers: up to twelve 16-bit and two 32-bit timers up to 168 MHz, each with up to 4 IC/OC/PWM or pulse counter and quadrature (incremental) encoder
- Up to 140 I/O ports with interrupt capability
- Up to 15 communication interfaces
  - 3 I2C interfaces (SMBus/PMBus)
  - 4 USARTs
  - 3 SPIs (37.5 Mbits/s
  - 2 CAN interfaces (2.0B Active)
  - SDIO interface
- Advanced connectivity
  - USB 2.0 full-speed device/host/OTG controller
  - 10/100 Ethernet MAC
- 8- to 14-bit parallel camera interface up to 54 Mbytes/s


CCM Data RAM 64 KB
NJTRST, JTDI
JTCK/SWCLK
JTDO/SWD, JTDO
JTAG & SW
MPU
ETM
NVIC
TRACECLK
TRACED[3:0]
ARM Cortex-M4F
168 MHz
FPU
D-BUS
I-BUS
S-BUS
MII or RMII as AF
MDIO as AF
Ethernet MAC
10/100
DMA/
FIFO
DP, DM
ULPI: CK, D(7:0), DIR, STP, NXT
SCL/SDA, INTN, ID, VBUS, SOF
PHY
USB
OTG HS
DMA/
FIFO
DMA2
8 Streams
FIFO
DMA1
8 Streams
FIFO
AHB bus-matrix 8S7M
AHB3
External memory
controller (FSMC)
SRAM, PSRAM, NOR Flash,
PC Card (ATA), NAND Flash
CLK, NE [3:0], A[23:0]
D[31:0], OEN, WEN,
NBL[3:0], NL, NREG
NWAIT/IORDY, CD
NIORD, IOWR, INT[2:3]
INTN, NIIS16 as AF
FIFO
FIFO
TDES, AES256
FIFO
HASH
RNG
ART ACCEL/
CACHE
Flash
up to
1MB
SRAM 112 KB
SRAM 16 KB
FIFO
Camera
interface
HSYNC, VSYNC
PIXCLK, D[13:0]
FIFO
USB
OTG FS
PHY
DP
DM
SCL, SDA, INTN, ID, VBUS, SOF
AHB2 168 MHz
AHB1 168 MHz
V_{DD12}
Power managmt
Voltage
regulator
3.3 V to 1.2 V
@VDD
V_{DD} = 1.8 to 3.6 V
V_{SS}
V_{CAP1}, V_{CAP2}
@VDDA
RC HS
RC LS
PLL1&2
POR
Reset
Int
Supply
supervision
POR/PDR/
BOR
PVD
V_{DDA}, V_{SSA}
NRST
@VDDA
@VDD
XTAL OSC
4-16 MHz
OSC_IN
OSC_OUT
IWDG
Reset &
clock
control
FCLK
HCLKx
PCLKx
Standby
interface
V_{BAT} = 1.65 to 3.6 V
@V_{BAT}
XTAL 32 kHz
OSC32_IN
OSC32_OUT
LS
LS
RTC
AWU
Backup register
4 KB BKPSRAM
RTC_AF1
RTC_AF1
PA[15:0]
GPIO PORT A
PB[15:0]
GPIO PORT B
PC[15:0]
GPIO PORT C
PD[15:0]
GPIO PORT D
PE[15:0]
GPIO PORT E
PF[15:0]
GPIO PORT F
PG[15:0]
GPIO PORT G
PH[15:0]
GPIO PORT H
PI[11:0]
GPIO PORT I
TIM2 32b
4 channels, ETR as AF
TIM3 16b
4 channels, ETR as AF
TIM4 16b
4 channels, ETR as AF
TIM5 32b
4 channels, ETR as AF
TIM12 16b
2 channels as AF
TIM13 16b
1 channel as AF
TIM14 16b
1 channel as AF
USART2 smcard irDA
RX, TX, CK,
CTS, RTS as AF
USART3 smcard irDA
RX, TX, CK
CTS, RTS as AF
UART4
RX, TX as AF
UART5
RX, TX as AF
SPI2/I2S2
MOSI/SD, MISO/SD_ext, SCK/CK
NSS/WS, MCK as AF
SPI3/I2S3
MOSI/SD, MISO/SD_ext, SCK/CK
NSS/WS, MCK as AF
I2C1/SMBUS
SCL, SDA, SMBA as AF
I2C2/SMBUS
SCL, SDA, SMBA as AF
I2C3/SMBUS
SCL, SDA, SMBA as AF
bxCAN1
TX, RX
bxCAN2
FIFO
TX, RX
APB1 42 MHz (max)
DMA2
DMA1
AHB/APB2
AHB/APB1
140 AF
EXT IT. WKUP
D[7:0]
CMD, CK as AF
SDIO / MMC
FIFO
4 compl. channels (TIM1_CH[1:4]N
4 channels (TIM1_CH[1:4]) ETR,
BKIN as AF
TIM1 / PWM 16b
4 compl. channels (TIM1_CH[1:4]N
4 channels (TIM1_CH[1:4]) ETR,
BKIN as AF
TIM8 / PWM 16b
2 channels as AF
TIM9 16b
1 channel as AF
TIM10 16b
1 channel as AF
TIM11 16b
RX, TX, CK,
CTS, RTS as AF
smcard irDA
USART 1
RX, TX, CK,
CTS, RTS as AF
smcard irDA
USART 6
MOSI, MISO
SCK, NSS as AF
SPI1
APB2 84 MHz
@VDDA
V_{DDREF_ADC}
Temperature sensor
8 analog inputs common
to the 3 ADCs
ADC1
8 analog inputs common
to the ADC1 & 2
ADC2
8 analog inputs to ADC3
ADC 3
IF
WWDG
TIM6 16b
TIM7 16b
@VDDA
DAC1
DAC2
ITF
DAC1_OUT
as AF
DAC2_OUT
as AF
ai18511b

2. Memory map:

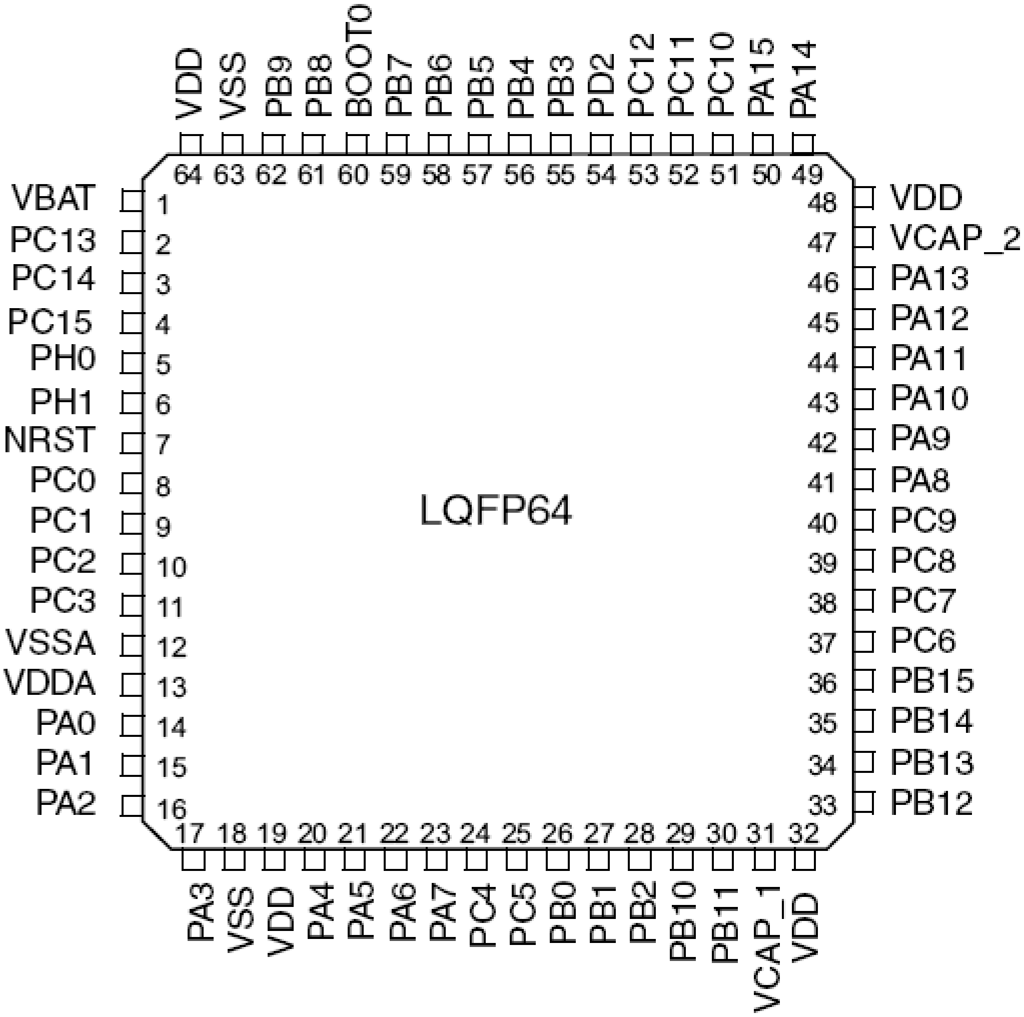

0xFFFF FFFF
512-Mbyte block 7 Cortex-M4's internal peripherals
0xE000 0000
0xDFFF FFFF
512-Mbyte block 6 Not used
0xC000 0000
0xBFFF FFFF
512-Mbyte block 5 FSMC registers
0xA000 0000
0x9FFF FFFF
512-Mbyte block 4 FSMC bank 3 & bank4
0x8000 0000
0x7FFF FFFF
512-Mbyte block 3 FSMC bank1 & bank2
0x6000 0000
0x5FFF FFFF
512-Mbyte block 2 Peripherals
0x4000 0000
0x3FFF FFFF
512-Mbyte block 1 SRAM
0x2000 0000
0x1FFF FFFF
512-Mbyte block 0 Code
0x0000 0000
Reserved
0x2002 0000 - 0x3FFF FFFF
SRAM (16 KB aliased by bit-banding)
0x2001 C000 - 0x2001 FFFF
SRAM (112 KB aliased by bit-banding)
0x2000 0000 - 0x2001 BFFF
Reserved
0x1FFF C008 - 0x1FFF FFFF
Option Bytes
0x1FFF C000 - 0x1FFF C007
Reserved
0x1FFF 7A10 - 0x1FFF 7FFF
System memory
0x1FFF 0000 - 0x1FFF 7A0F
Reserved
0x1001 0000 - 0x1FFE FFFF
TCM data RAM (64 KB data SRAM)
0x1000 0000 - 0x1000 FFFF
Reserved
0x0810 0000 - 0x0FFF FFFF
Flash
0x0800 0000 - 0x080F FFFF
Reserved
0x0010 0000 - 0x07FF FFFF
Aliased to Flash, system memory or SRAM depending on the BOOT pins
0x0000 0000 - 0x000F FFFF
Reserved
0xA000 1000 - 0xBFFF FFFF
AHB3
0xA000 0FFF
0x6000 0000
Reserved
0x5006 0C00 - 0x5FFF FFFF
AHB2
0x5006 0BFF
0x5006 0000
Reserved
0x5005 0400 - 0x5005 FFFF
AHB2
0x5005 03FF
0x5005 0000
Reserved
0x5004 0000 - 0x5004 FFFF
AHB2
0x5003 FFF
0x5000 0000
Reserved
0x4002 9400 - 0x4FFF FFFF
AHB1
0x4004 0000 - 0x4007 FFFF
0x4002 3C00
Reset clock controller (RCC)
0x4002 3800 - 0x4002 3BFF
Reserved
0x4002 3400 - 0x4002 37FF
CRC
0x4002 3000 - 0x4002 33FF
Reserved
0x4002 2400 - 0x4002 2FFF
AHB1
0x4002 23FF
0x4002 000
Reserved
0x4001 4C00 - 0x4001 FFFF
APB2
0x4001 4BFF
0x4001 4000
EXTI
0x4001 3C00 - 0x4001 3FFF
SYSCFG
0x4001 3800 - 0x4001 3BFF
Reserved
0x4001 3400 - 0x4001 37FF
APB2
0x4001 33FF
0x4001 0000
Reserved
0x4000 7800 - 0x4000 FFFF
APB1
0x4000 77FF
0x4000 2800
Reserved
0x4000 2400 - 0x4000 27FF
APB1
0x4000 23FF
0x4000 0000
ai18513c

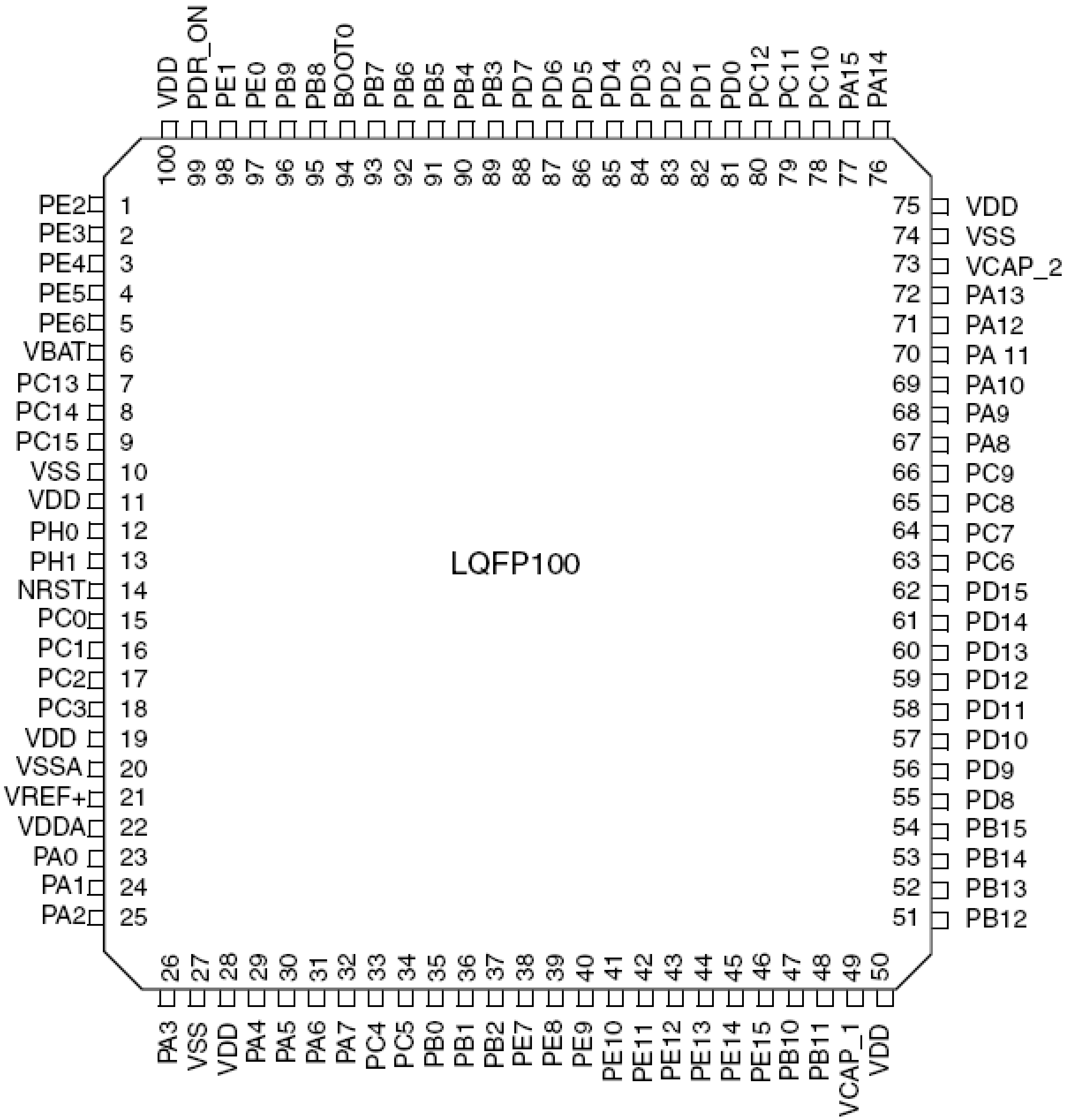
- Multi-AHB Matrix



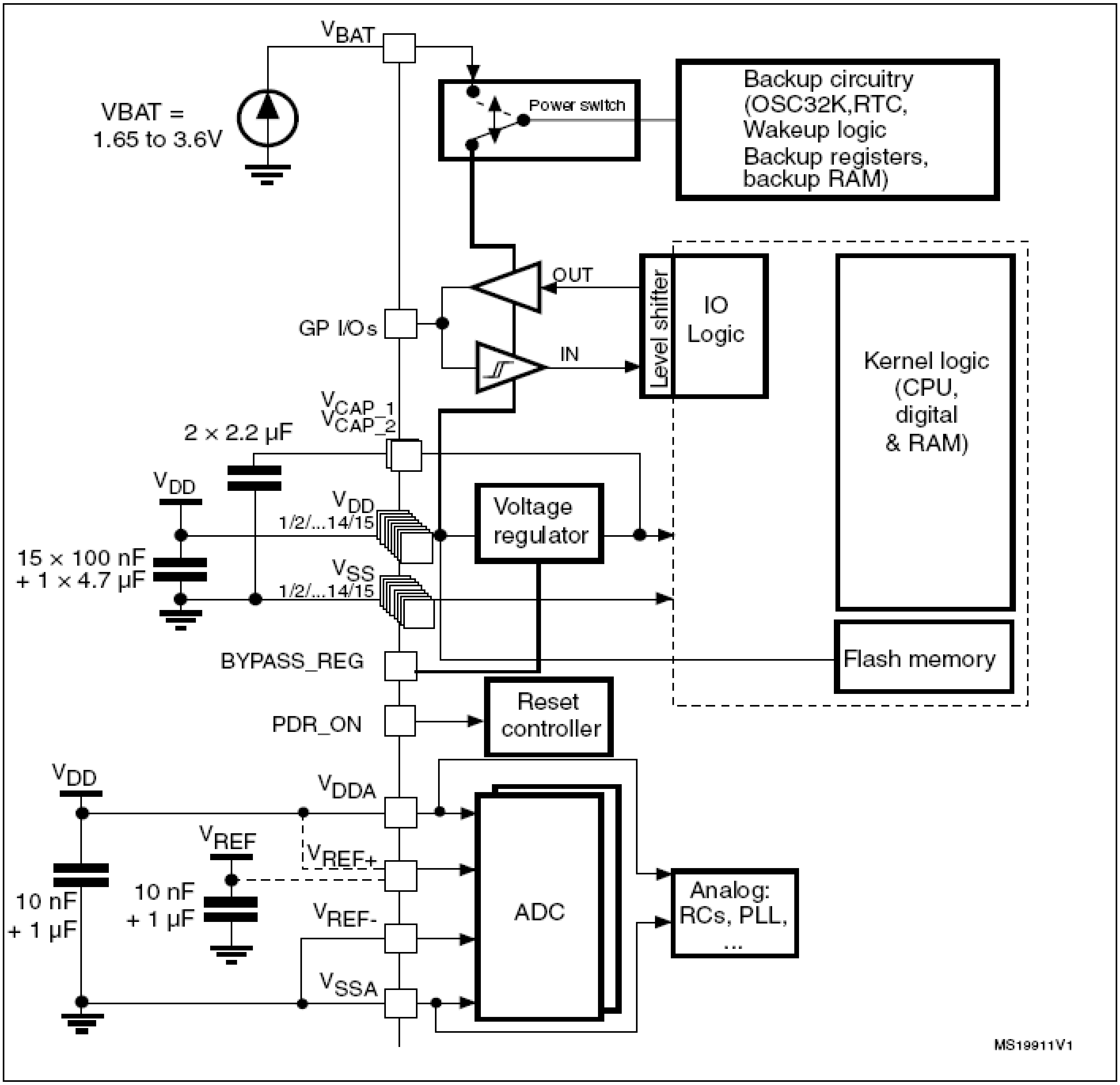
+ S0: I-bus: is used by the core to fetch instructions.
+ S1: D-bus is used by the core for literal load and debug access.
+ S2: S-bus is used to access data located in a peripheral or in SRAM. Instructions may also be fetch on this bus (less efficient than ICode).
+ S3, S4: DMA memory bus is used by the DMA to perform transfer memories.
+ **S5: DMA peripheral bus** is used by the DMA to access AHB peripherals or to perform memory-to-memory transfers.
+ S6: Ethernet DMA bus is used by the Ethernet DMA to load/store data
+ S7: USB OTG HS DMA bus is used by the USB OTG DMA to load/store data
+ AHB/APB bridges (APB): 2 AHB/APB bridges, APB1 and APB2, provide full synchronous connections between the AHB and the two APB buses, allowing flexible selection of the peripheral frequency.

3. Pinouts:


VDD
VSS
PB9
PB8
BOOT0
PB7
PB6
PB5
PB4
PB3
PD2
PC12
PC11
PC10
PA15
PA14
64
63
62
61
60
59
58
57
56
55
54
53
52
51
50
49
VBAT
1
PC13
2
PC14
3
PC15
4
PH0
5
PH1
6
NRST
7
PC0
8
PC1
9
PC2
10
PC3
11
VSSA
12
VDDA
13
PA0
14
PA1
15
PA2
16
48
VDD
47
VCAP_2
46
PA13
45
PA12
44
PA11
43
PA10
42
PA9
41
PA8
40
PC9
39
PC8
38
PC7
37
PC6
36
PB15
35
PB14
34
PB13
33
PB12
LQFP64
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
PA3
VSS
VDD
PA4
PA5
PA6
PA7
PC4
PC5
PB0
PB1
PB2
PB10
PB11
VCAP_1
VDD

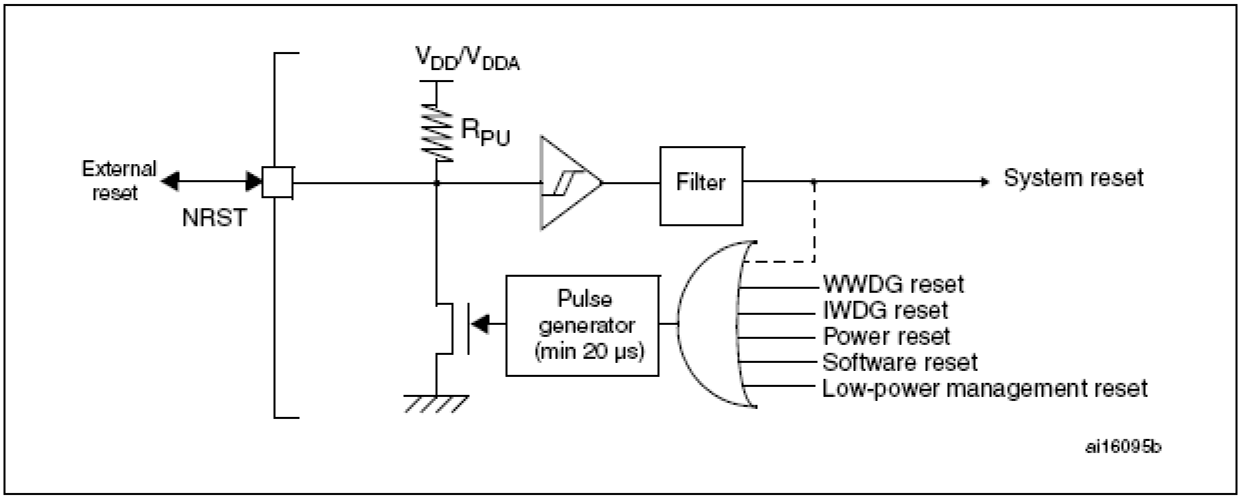

LQFP100
VDD 100
PDR_ON 99
PE1 98
PE0 97
PB9 96
PB8 95
BOOT0 94
PB7 93
PB6 92
PB5 91
PB4 90
PB3 89
PD7 88
PD6 87
PD5 86
PD4 85
PD3 84
PD2 83
PD1 82
PD0 81
PC12 80
PC11 79
PC10 78
PA15 77
PA14 76
PE2 1
PE3 2
PE4 3
PE5 4
PE6 5
VBAT 6
PC13 7
PC14 8
PC15 9
VSS 10
VDD 11
PH0 12
PH1 13
NRST 14
PC0 15
PC1 16
PC2 17
PC3 18
VDD 19
VSSA 20
VREF+ 21
VDDA 22
PA0 23
PA1 24
PA2 25
75 VDD
74 VSS
73 VCAP_2
72 PA13
71 PA12
70 PA 11
69 PA10
68 PA9
67 PA8
66 PC9
65 PC8
64 PC7
63 PC6
62 PD15
61 PD14
60 PD13
59 PD12
58 PD11
57 PD10
56 PD9
55 PD8
54 PB15
53 PB14
52 PB13
51 PB12
26 PA3
27 VSS
28 VDD
29 PA4
30 PA5
31 PA6
32 PA7
33 PC4
34 PC5
35 PB0
36 PB1
37 PB2
38 PE7
39 PE8
40 PE9
41 PE10
42 PE11
43 PE12
44 PE13
45 PE14
46 PE15
47 PB10
48 PB11
49 VCAP_1
50 VDD

4. Power control:



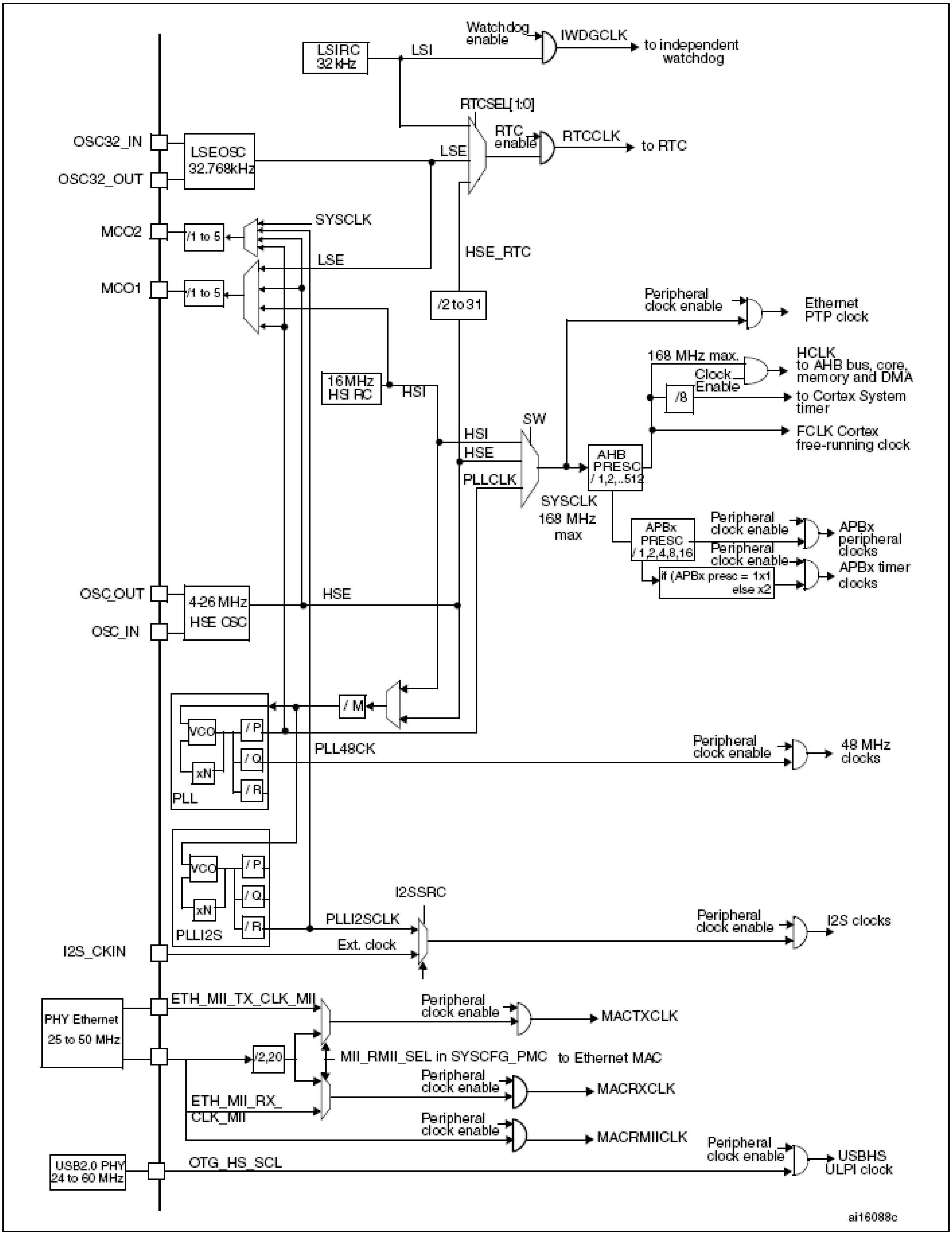
- The device requires a 1.8-to-3.6 V operating voltage supply (VDD). An embedded linear voltage regulator is used to supply the internal 1.2 V digital power.
- The real-time clock (RTC), the RTC backup registers, and the backup SRAM (BKP SRAM) can be powered from the VBAT voltage when the main VDD supply is powered off.

5. Reset and clock control:

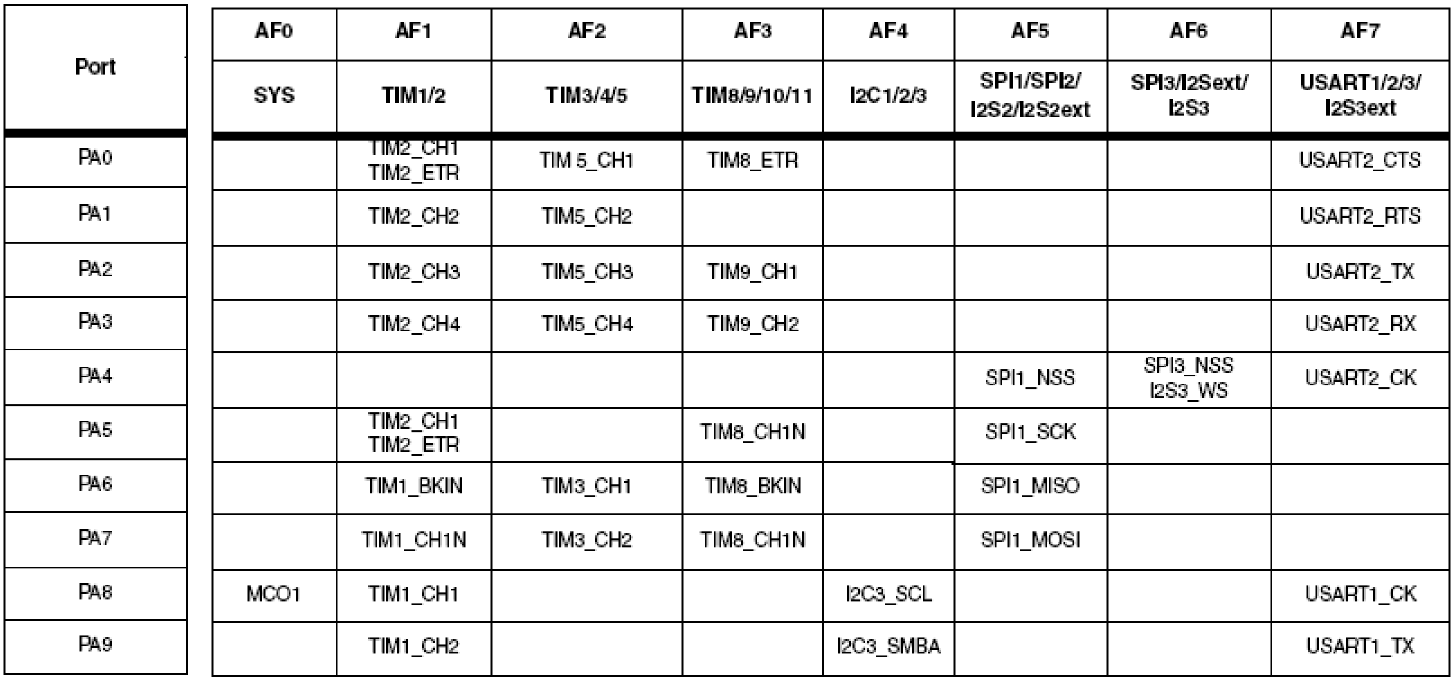


- A system reset is generated when one of the following events occurs:

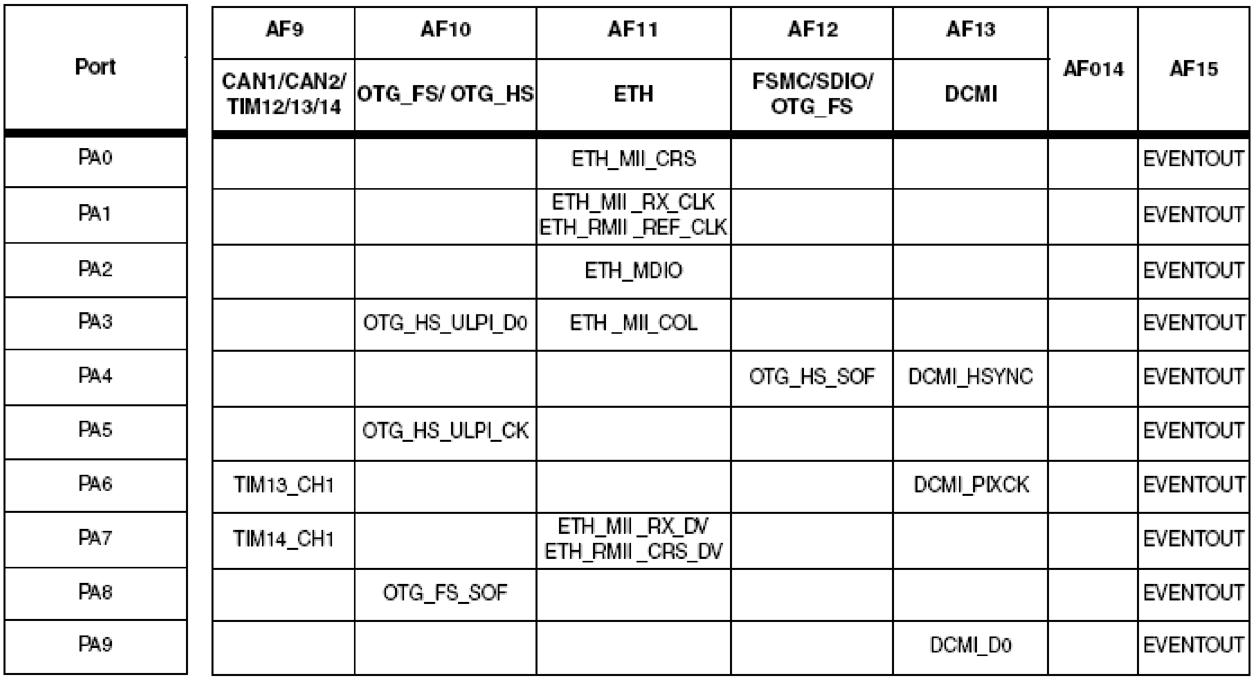
1. A low level on the NRST pin (external reset)
2. Window watchdog end of count condition (WWDG reset)
3. Independent watchdog end of count condition (IWDG reset)
4. A software reset (SW reset) (see *Software reset*)
5. Low-power management reset (see *Low-power management reset*)


LSIRC
32 kHz
LSI
Watchdog enable
IWDGCLK
to independent watchdog
RTCSEL[1:0]
OSC32_IN
OSC32_OUT
LSEOSC
32.768kHz
LSE
RTC enable
RTCCLK
to RTC
MCO2
/1 to 5
SYSCLK
LSE
MCO1
/1 to 5
HSE_RTC
/2 to 31
Peripheral clock enable
Ethernet PTP clock
168 MHz max.
Clock Enable
HCLK
to AHB bus, core, memory and DMA
/8
to Cortex System timer
FCLK Cortex free-running clock
16MHz
HSI RC
HSI
SW
HSI
HSE
PLLCLK
AHB PRESC / 1,2,..512
SYSCLK
168 MHz max
APBx PRESC / 1,2,4,8,16
Peripheral clock enable
APBx peripheral clocks
Peripheral clock enable
if (APBx presc = 1x1 else x2
APBx timer clocks
OSC_OUT
OSC_IN
4-26 MHz
HSE OSC
HSE
/ M
VCO
xN
/ P
/ Q
/ R
PLL
PLL48CK
Peripheral clock enable
48 MHz clocks
VCO
xN
/ P
/ Q
/ R
PLLI2S
I2SSRC
PLLI2SCLK
Ext. clock
I2S_CKIN
Peripheral clock enable
I2S clocks
PHY Ethernet
25 to 50 MHz
ETH_MII_TX_CLK_MII
Peripheral clock enable
MACTXCLK
/2,20
MII_RMII_SEL in SYSCFG_PMC
to Ethernet MAC
Peripheral clock enable
MACRXCLK
ETH_MII_RX_CLK_MII
Peripheral clock enable
MACRMIICLK
USB2.0 PHY
24 to 60 MHz
OTG_HS_SCL
Peripheral clock enable
USBHS ULPI clock
ai16088c

6. General purpose IOs (GPIO)



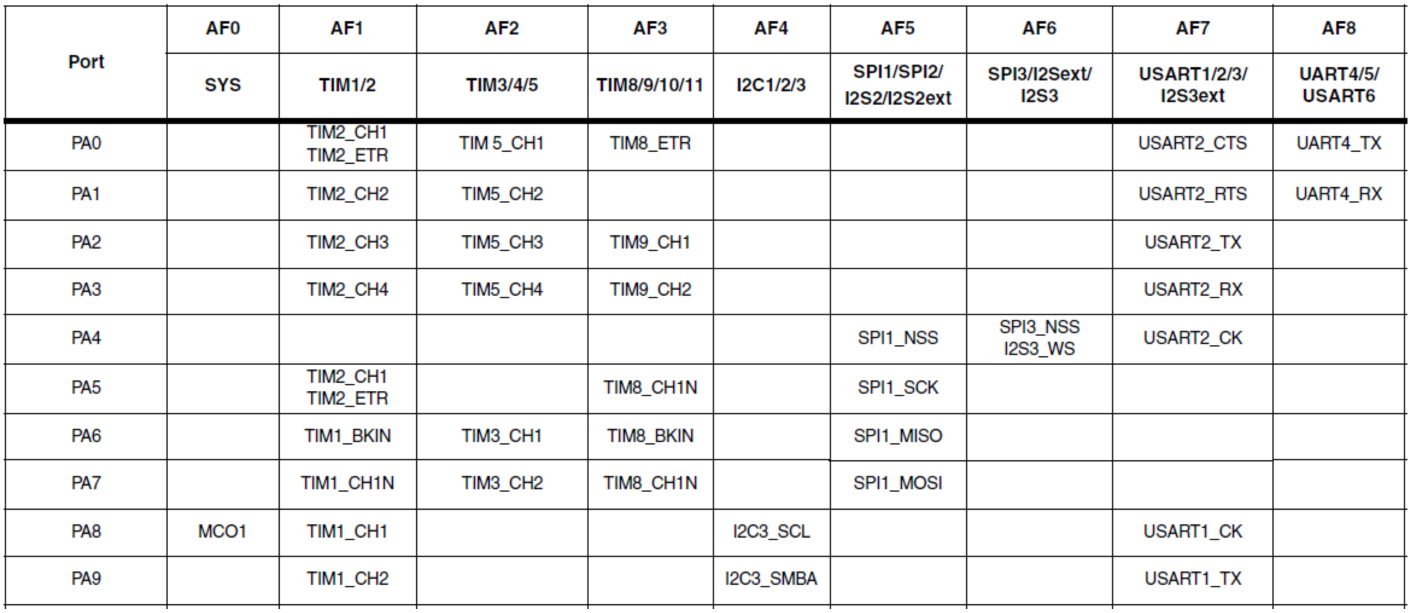
Port	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7
	SYS	TIM1/2	TIM3/4/5	TIM8/9/10/11	I2C1/2/3	SPI1/SPI2/ I2S2/I2S2ext	SPI3/I2Sext/ I2S3	USART1/2/3/ I2S3ext
PA0		TIM2_CH1 TIM2_ETR	TIM 5_CH1	TIM8_ETR				USART2_CTS
PA1		TIM2_CH2	TIM5_CH2					USART2_RTS
PA2		TIM2_CH3	TIM5_CH3	TIM9_CH1				USART2_TX
PA3		TIM2_CH4	TIM5_CH4	TIM9_CH2				USART2_RX
PA4						SPI1_NSS	SPI3_NSS I2S3_WS	USART2_CK
PA5		TIM2_CH1 TIM2_ETR		TIM8_CH1N		SPI1_SCK		
PA6		TIM1_BKIN	TIM3_CH1	TIM8_BKIN		SPI1_MISO		
PA7		TIM1_CH1N	TIM3_CH2	TIM8_CH1N		SPI1_MOSI		
PA8	MCO1	TIM1_CH1			I2C3_SCL			USART1_CK
PA9		TIM1_CH2			I2C3_SMBA			USART1_TX



Port	AF9	AF10	AF11	AF12	AF13	AF014	AF15
	CAN1/CAN2/ TIM12/13/14	OTG_FS/ OTG_HS	ETH	FSMC/SDIO/ OTG_FS	DCMI		
PA0			ETH_MII_CRS				EVENTOUT
PA1			ETH_MII _RX_CLK ETH_RMII _REF_CLK				EVENTOUT
PA2			ETH_MDIO				EVENTOUT
PA3		OTG_HS_ULPI_D0	ETH _MII_COL				EVENTOUT
PA4				OTG_HS_SOF	DCMI_HSYNC		EVENTOUT
PA5		OTG_HS_ULPI_CK					EVENTOUT
PA6	TIM13_CH1				DCMI_PIXCK		EVENTOUT
PA7	TIM14_CH1		ETH_MII _RX_DV ETH_RMII _CRS_DV				EVENTOUT
PA8		OTG_FS_SOF					EVENTOUT
PA9					DCMI_D0		EVENTOUT

- IO port control registers: GPIOx_MODER, GPIOx_OTYPER, GPIOx_OSPEEDR, GPIOx_PUPDR (x = A-I)
- I/O port data registers: GPIOx_IDR, GPIOx_ODR, GPIOx_BSRR
- I/O function: GPIOx_AFRL, GPIO_AFRH

Table 6. Alternate function mapping



Port	AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	AF8
	SYS	TIM1/2	TIM3/4/5	TIM8/9/10/11	I2C1/2/3	SPI1/SPI2/ I2S2/I2S2ext	SPI3/I2Sext/ I2S3	USART1/2/3/ I2S3ext	UART4/5/ USART6
PA0		TIM2_CH1 TIM2_ETR	TIM 5_CH1	TIM8_ETR				USART2_CTS	UART4_TX
PA1		TIM2_CH2	TIM5_CH2					USART2_RTS	UART4_RX
PA2		TIM2_CH3	TIM5_CH3	TIM9_CH1				USART2_TX	
PA3		TIM2_CH4	TIM5_CH4	TIM9_CH2				USART2_RX	
PA4						SPI1_NSS	SPI3_NSS I2S3_WS	USART2_CK	
PA5		TIM2_CH1 TIM2_ETR		TIM8_CH1N		SPI1_SCK			
PA6		TIM1_BKIN	TIM3_CH1	TIM8_BKIN		SPI1_MISO			
PA7		TIM1_CH1N	TIM3_CH2	TIM8_CH1N		SPI1_MOSI			
PA8	MCO1	TIM1_CH1			I2C3_SCL			USART1_CK	
PA9		TIM1_CH2			I2C3_SMBA			USART1_TX	

Ví dụ: Cấu hình các chức năng:

UART:	UART2:	PA2, PA3
SPI:	SPI1:	PA4 – PA7
DI:		PA8
DO:		PA9

1. Cho phép xung clock:

```
RCC_AHB1ENR |=        // Cho phép Port A
                      // GPIOAEN = 1
RCC_APB1ENR |=        // UART2, SPI1
RCC_APB2ENR |=        // UART2, SPI1
```

2. Cấu hình chức năng ngoại vi:

```
GPIOA_MODER |=
GPIOA_AFRH |=         // UART2 AF
GPIOA_AFRL |=         // SPI1 AF
```

3. Cấu hình điện trở, lọc ngõ ra:

```
GPIOA_OTYPER |=
GPIOA_OSPEEDR |=
GPIOA_PUPDR |=
```

* Xuất giá trị ra port
- Thanh ghi GPIOA_ODR

```
  // PA9 = 1
  GPIOA_ODR =
  // PA9 = 0
  GPIOA_ODR =
```

- Thanh ghi GPIOA_BSRR

```
  // PA9 = 1
  GPIOA_BSRR =
  //PD9 = 0
  GPIOA_BSRR =
```

* Đọc port PA8
- Thanh ghi GPIOA_IDR

```
  // Đọc cả Port A
  a = GPIOA_IDR;

// Đọc PA8
  a = GPIOA_IDR
```