ISEN
ALL IS DIGITAL!
yncréa

ARM Cortex-M Introduction

V1.0
10/02/2018

Agenda

1. Market Overview
2. ARM Families
3. Keil debugger
4. Cortex-M Architecture

CORTEX M Market Overview

ISEN ALL IS DIGITAL! | yncréa

- ✓Cortex is a core for microcontrollers designed by ARM (British company bought by the Japanese company Softbank for 27 B$)
- ✓ARM does not sell any silicon. Its business model is based on license to silicon vendors
- ✓Leader for 32-bit solution

Major vendors of Cortex Microcontrollers

ISEN ALL IS DIGITAL! | yncréa

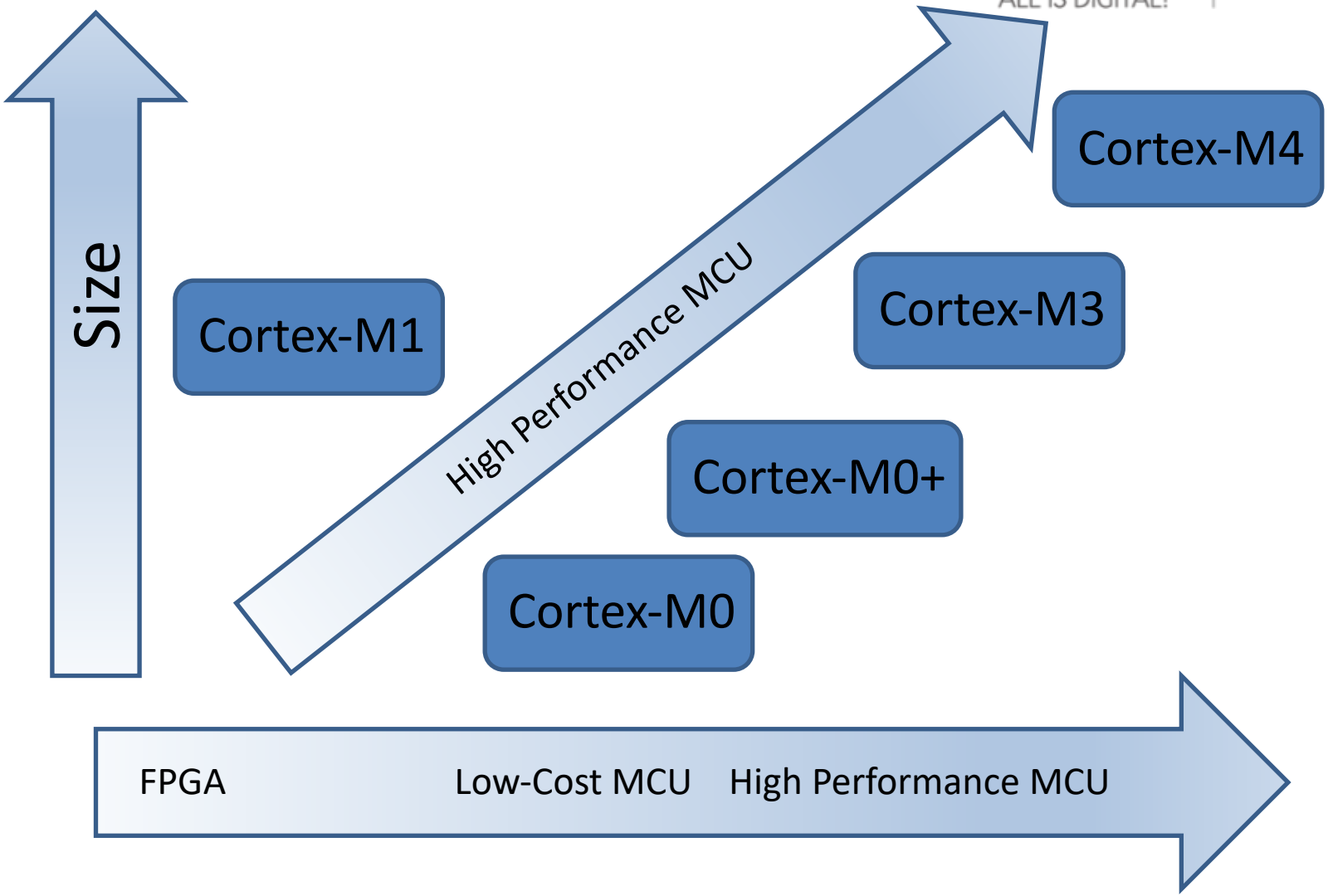
- ✓STMicroelectronics
- ✓TI
- ✓Renesas
- ✓Microchip
- ✓ATMEL
- ✓CYPRESS
- ✓Infineon
- ✓….

Most (if not all) microcontroller vendors have a Cortex M Line

CORTEX Profiles

- ✓ Cortex-A: designed for high end applications (servers, set top boxes, mobile applications
  - ✓ Capable of running feature-rich operating systems like Linux
- ✓ Cortex-R: Real time profile that delivers a high-performance processor. Very often a Cortex-R processor forms a part of a « system-on-chip » design that is focused on a specific task such as hard disk drive
- ✓ Cortex-M: designed to be used within a small microcontroller

Cortex-M Profile



Comparison Clock cycles

Table 1.1: Number of Cycles Taken for a 16 × 16 Multiply against Typical 8- and 16-bit Architectures

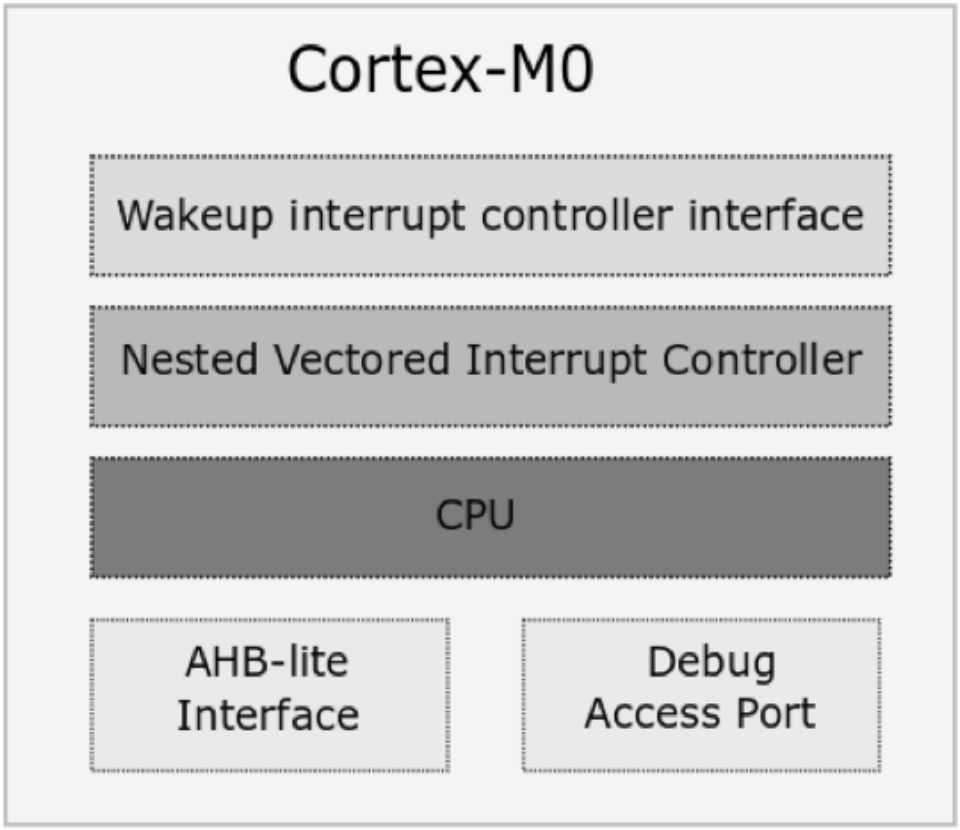
8-Bit Example (8051)	16-Bit Example	ARM Cortex-M

MOV A, XL ; 2 bytes
MOV B, YL ; 3 bytes
MUL AB; 1 byte
MOV R0, A; 1 byte
MOV R1, B; 3 bytes
MOV A, XL ; 2 bytes
MOV B, YH ; 3 bytes
MUL AB; 1 byte
ADD A, R1; 1 byte
MOV R1, A; 1 byte
MOV A, B ; 2 bytes
ADDC A, #0 ; 2 bytes
MOV R2, A; 1 byte
MOV A, XH ; 2 bytes
MOV B, YL ; 3 bytes
MUL AB; 1 byte
ADD A, R1; 1 byte
MOV R1, A; 1 byte
MOV A, B ; 2 bytes
ADDC A, #0 ; 2 bytes
MOV R3, A; 1 byte
	MOV R1,&MulOp1
	MOV R2,&MulOp2
	MOV SumLo,R3
	MOV SumHi,R4
	
	(Memory mapped multiply unit)
		MULS r0,r1,r0

Time: 48 clock cycles* **Code size:** 48 bytes	**Time:** 8 clock cycles **Code size:** 8 bytes	**Time:** 1 clock cycle **Code size:** 2 bytes

* *cycle count for a single cycle 8051 processor.*

Cortex-M0 block diagram

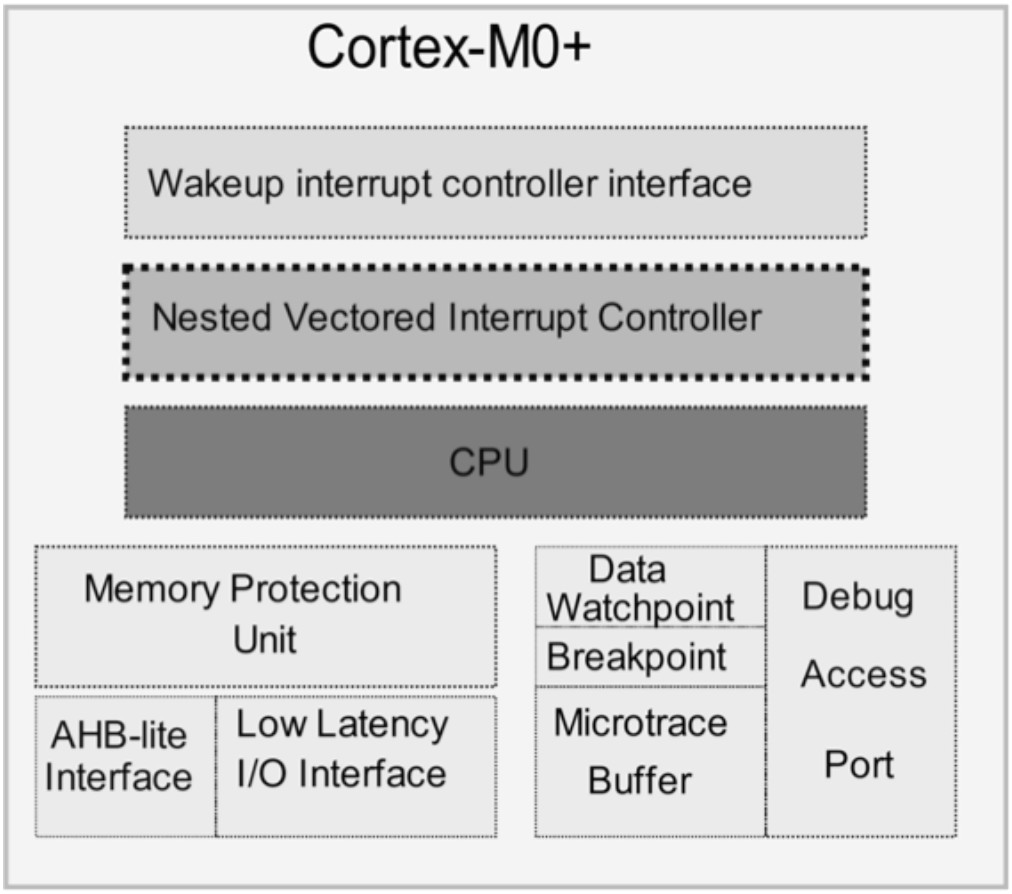


A few words about Cortex-M0

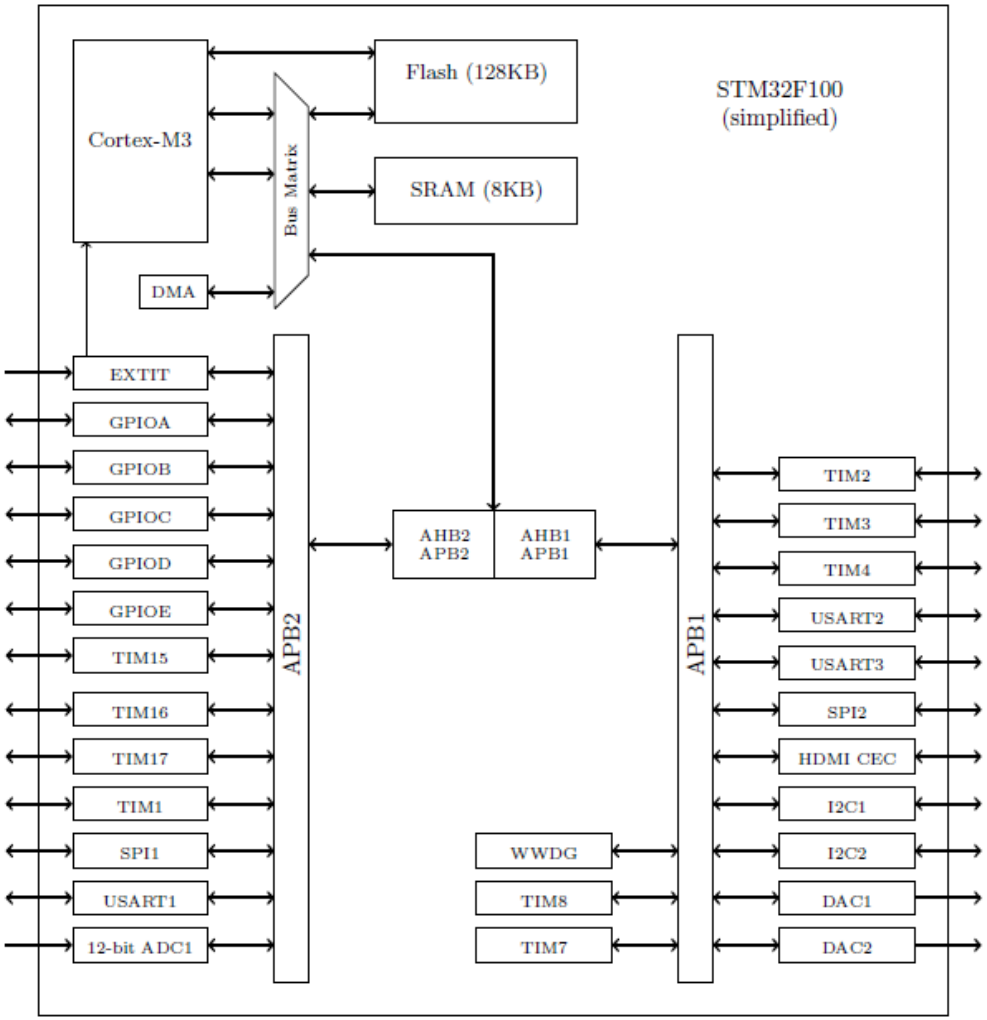
ISEN ALL IS DIGITAL! | yncréa

- ✓ The Cortex-M0 is a much smaller processor than the cortex-M3.
- ✓ Can be as small as 12 k gates
- ✓ Cortex-M0 is typically designed into microcontrollers that are intended to be low-cost devices and/or low power operation.
- ✓ Cortex-M0+ has a complete set compatibility with the cortex-M0 allowing to use the same compiler and debug tools.
- ✓ Major advantage of the Cortex-M0+ is its power consumption, which is 11 µW/MHz compared to 16 µW/MHz for the Cortex-M0 and 32 µW/MHz for the Cortex-M3.

Cortex-M0+ block diagram



Cortex-M3 block diagram



Cortex-M4 block diagram

Cortex M4 has all the DSP functions and FPU registers

Core Architecture

specific hardware is through three memory buses – ICode, DCode, and System – which are defined to access different regions of memory.

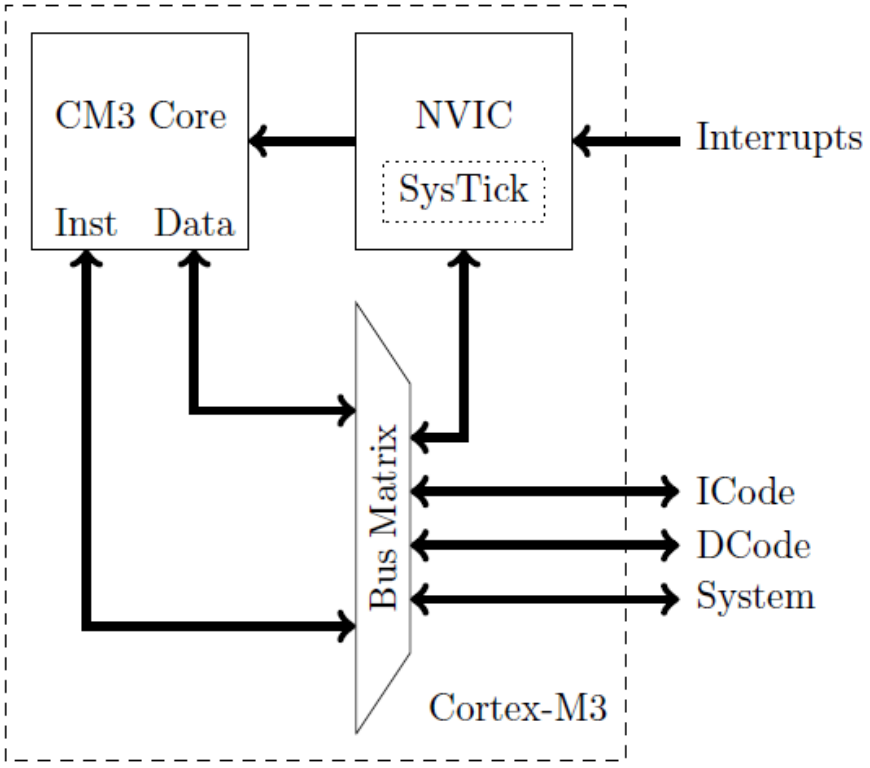


Figure 2.2: Simplified Cortex-M3 Core Architecture

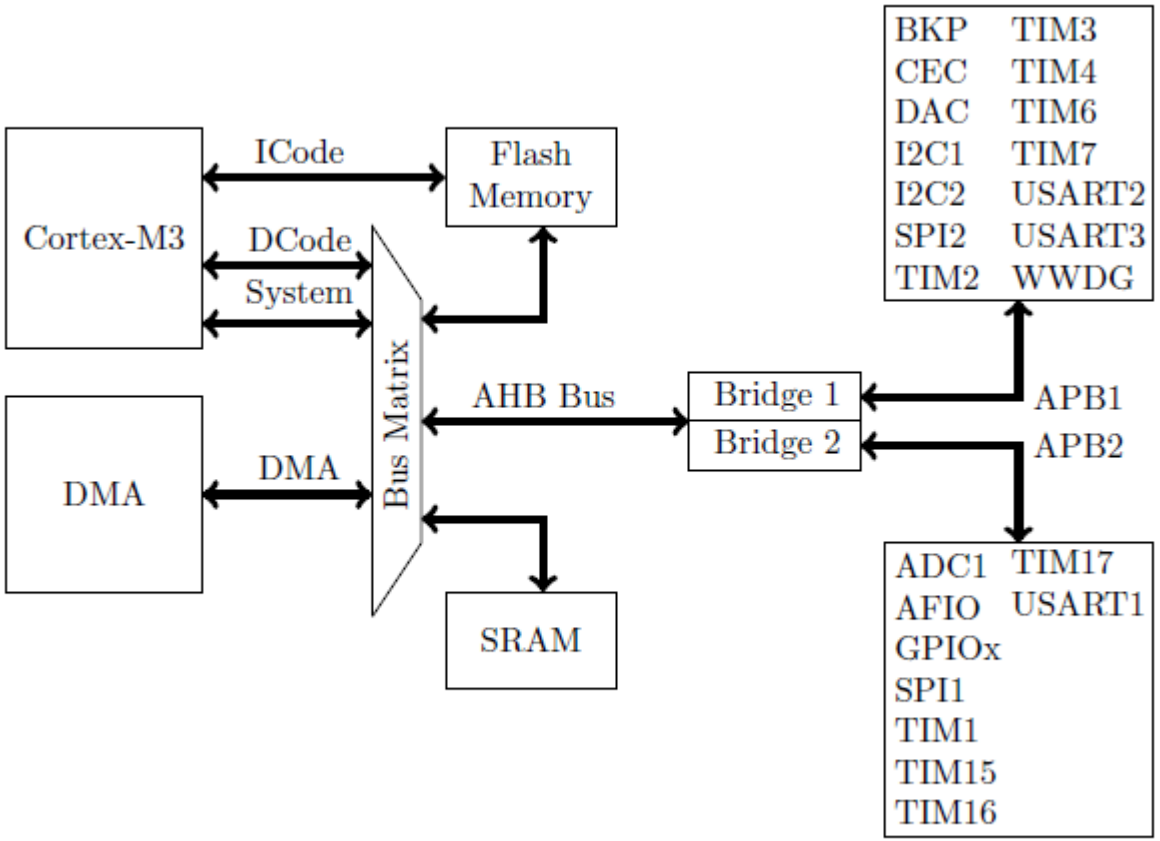
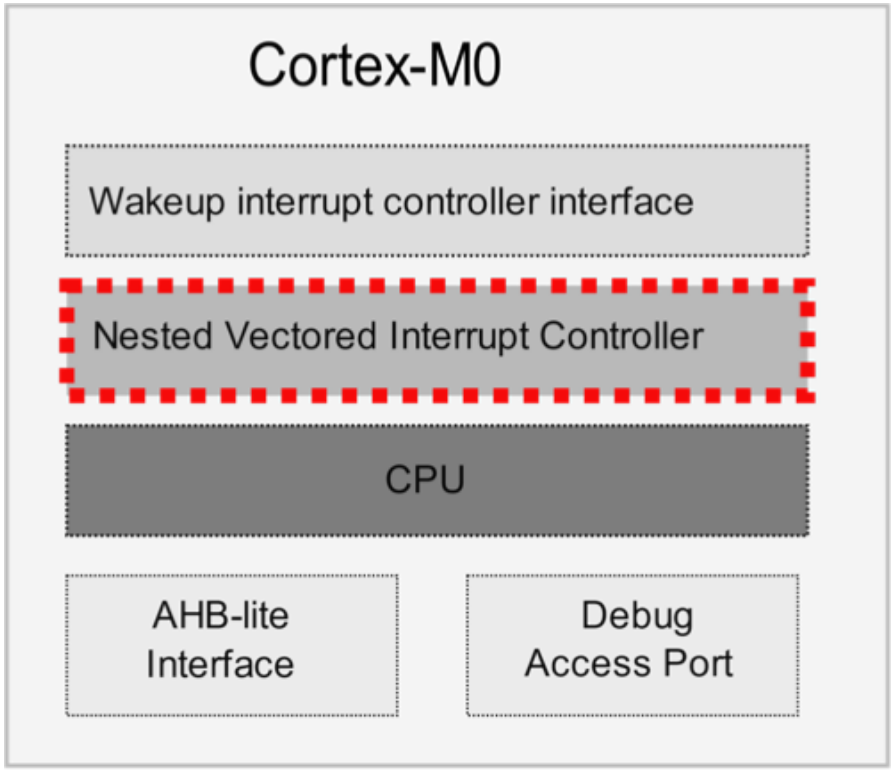


Figure 2.3: STM32 Medium Density Value-Line Bus Architecture

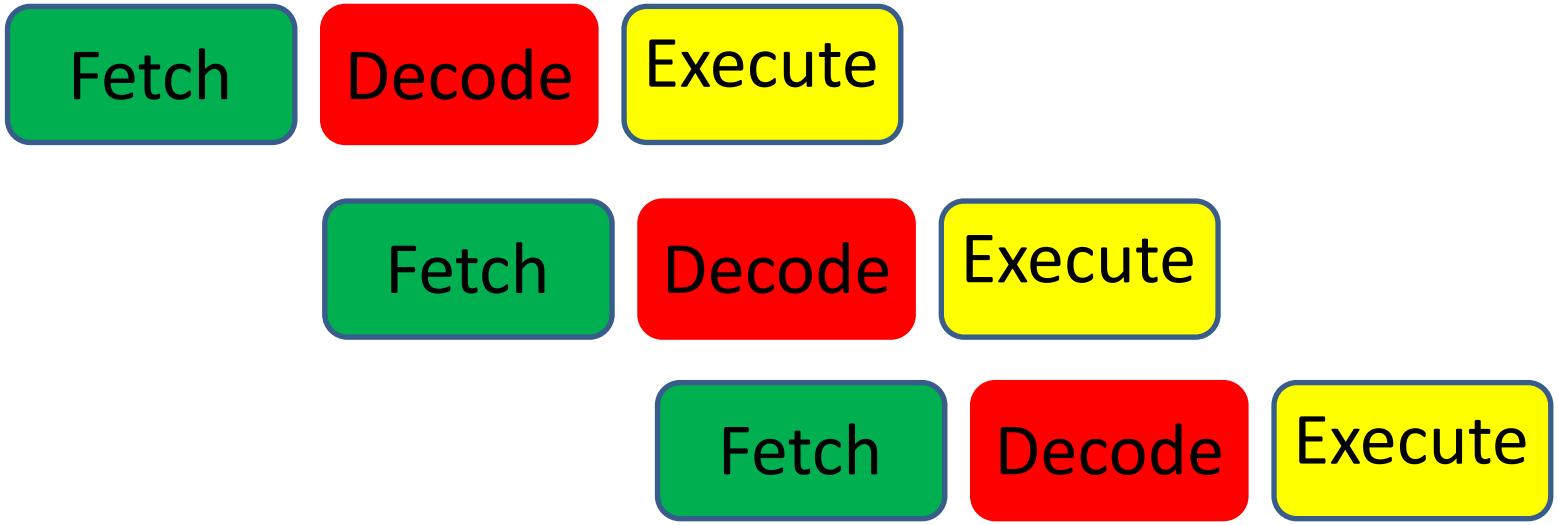
NVIC


Cortex-M0
Wakeup interrupt controller interface
Nested Vectored Interrupt Controller
CPU
AHB-lite Interface
Debug Access Port

CORTEX-M3/4 Three-Stage Pipeline

- ✓ The Cortex-M3/4 is a Reduced Instruction Set Computer (RISC) processor where most instructions will execute in a single cycle
- ✓ To perform this task the Cortex M3/4 has a 3-stage pipeline with branch prediction
- ✓ The 3 stages are called: Fetch, Decode, Execute

Code Execution in Cortex-M3/M4



- ✓ While one instruction is being executed, a second one is being decoded and a third one is being fetched
- ✓ Great when the code is going in straight line
- ✓ When the program branches, the pipeline must be flushed and refilled with new instructions

Cortex-M0/M0+/M3/M4 Pipelines

ISEN ALL IS DIGITAL! | yncréa

- ✓ Cortex-M3/M4 have a 3-stage pipeline with branch prediction
- ✓ Cortex-M0 has a 3-stage pipeline without a branch prediction
- ✓ Cortex-M0+ has a 2-stage pipeline (Fetch and decode in the same stage)
- ✓ When Cortex-M0/M0+ executes a conditional branch, the instructions in the pipeline are no longer valid (pipeline must be flushed everytime there is a branch)

Cortex-M0/M0+/M3/M4 Pipelines

With no branch prediction, the pipeline must be refilled every time there is a branch.

- ✓ Obvious impact on performance
- ✓ Also means flash memory more often and each access costs energy as well as time
- ✓ By moving to a 2-stage pipeline for Cortex-M0+, the number of flash memory accesses and hence the runtime energy consumption is also reduced.

ISEN | école d'ingénieurs
ALL IS DIGITAL!

CORTEX-M Tutorial

Debugger tools - Keil

Setup 1/3

MDK-Lite version
http://www2.keil.com/mdk5/install

Pack STMicroelectronics for STM32F4x series, STM32L4xx series & STM32NUCLEO_BSP.

STM32CUBEMX
http://www.st.com/content/st_com/en/products/development-tools/software-development-tools/stm32-software-development-tools/stm32-configurators-and-code-generators/stm32cubemx.html

Setup 2/3

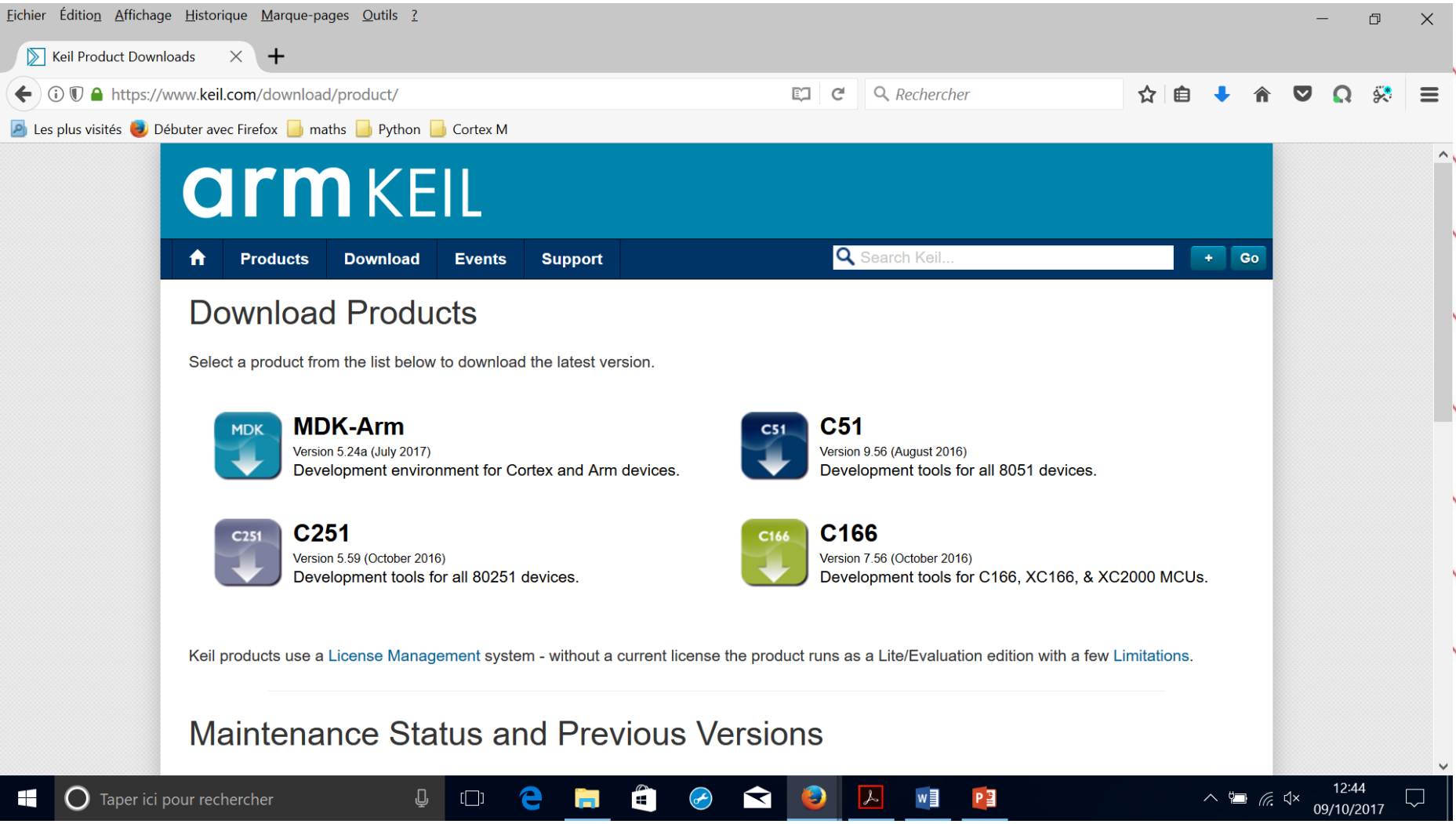
Examples
http://www.st.com/content/st_com/en/products/embedded-software/mcus-embedded-software/stm32-embedded-software/stm32cube-embedded-software/stm32cubef4.html

Standard peripheral lib
http://www.st.com/content/st_com/en/products/embedded-software/mcus-embedded-software/stm32-embedded-software/stm32-standard-peripheral-libraries/stsw-stm32065.html

Setup 3/3

ST-Link/V2
http://www.st.com/content/st_com/en/products/development-tools/hardware-development-tools/development-tool-hardware-for-mcus/debug-hardware-for-mcus/debug-hardware-for-stm32-mcus/st-link-v2.html

Download uKeil



µVision Lite Limitations

ISEN ALL IS DIGITAL! | yncréa

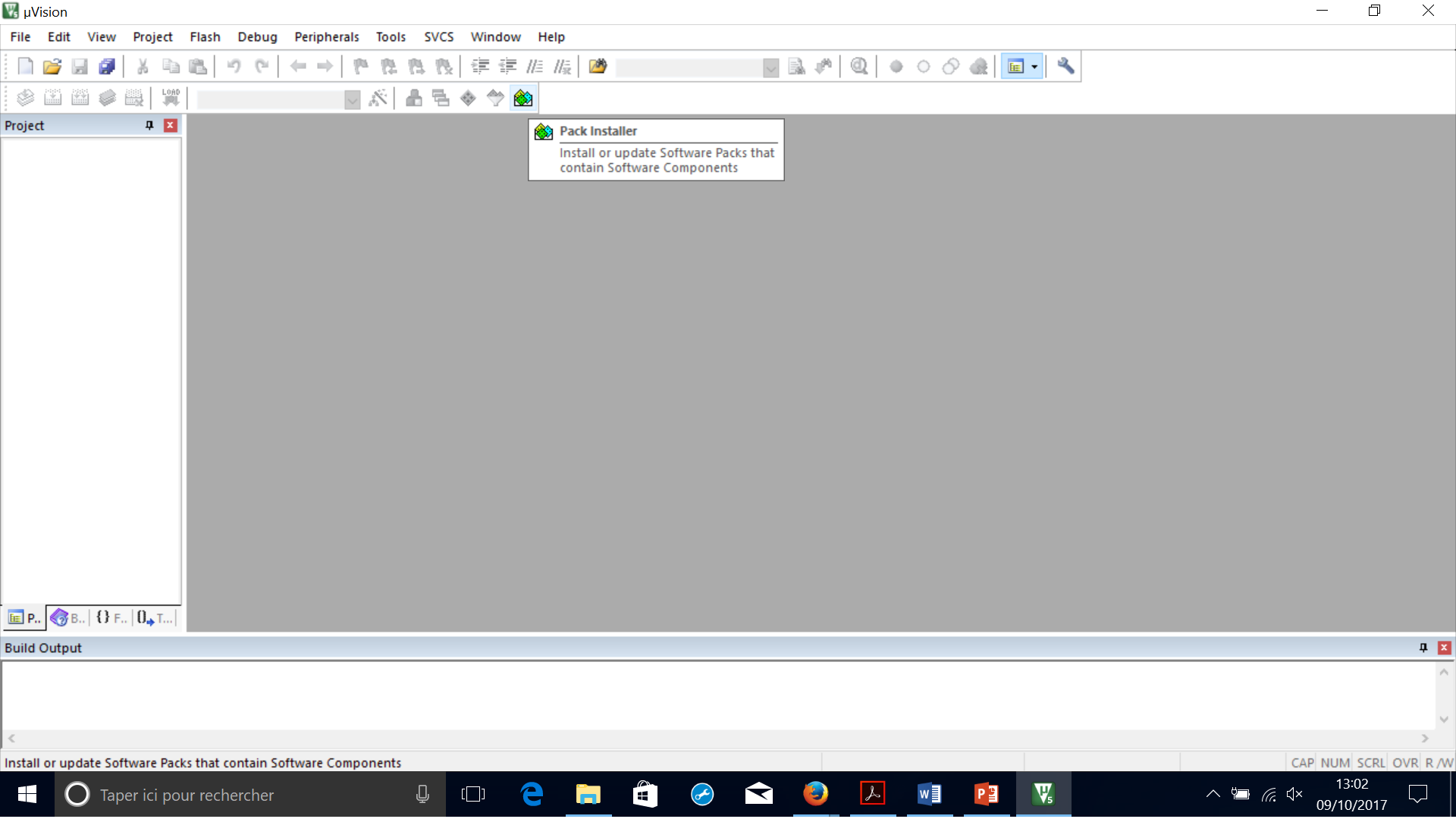
Limitations

Keil development development tools without a current product license run as a Lite/Evaluation edition and have the following restrictions:

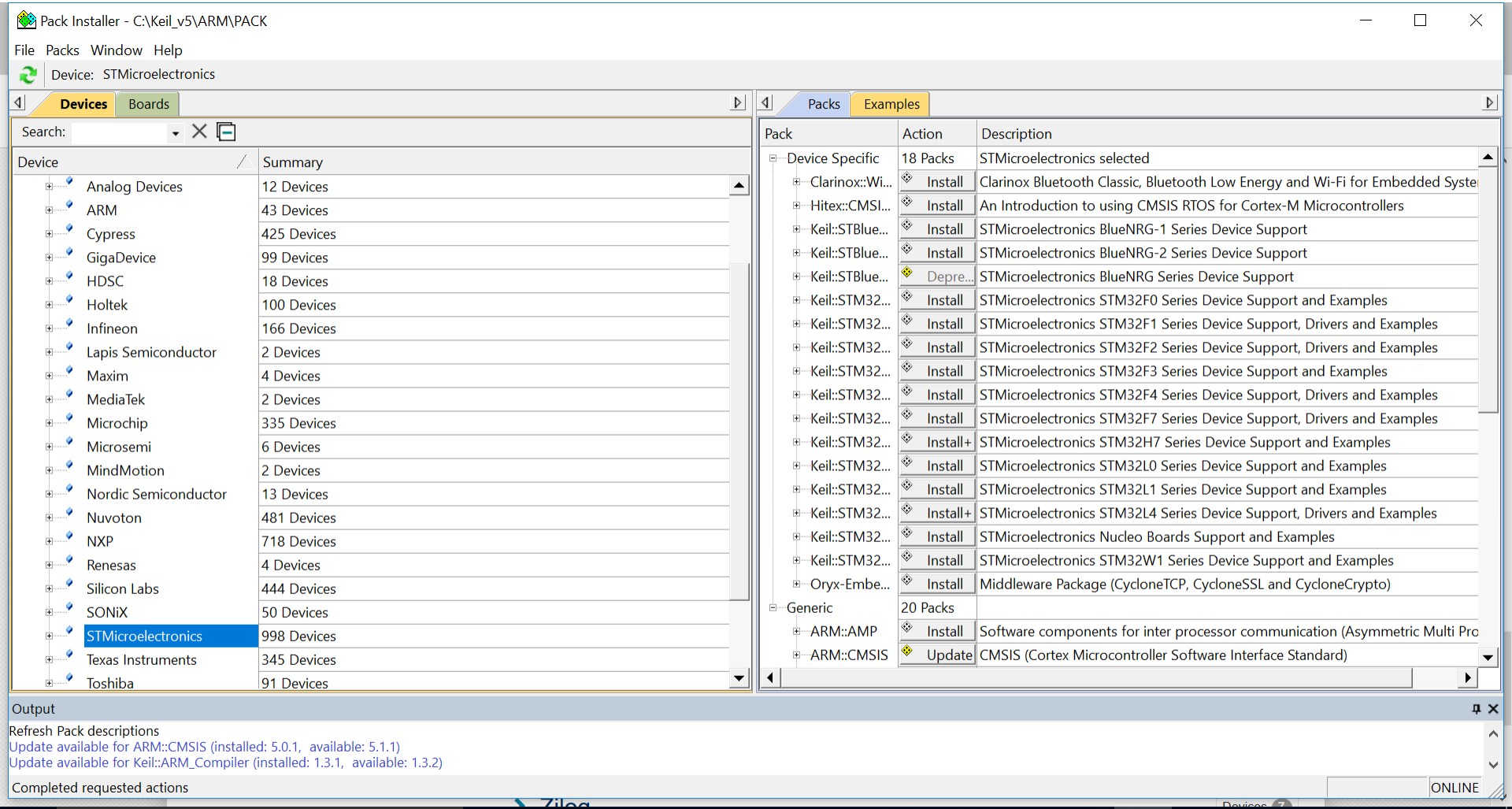
MDK-ARM Lite Edition

- Programs that generate more than 32 Kbytes of code and data will not compile, assemble, or link.
- The debugger supports programs that are 32 Kbytes or smaller.
- The compiler does not generate a disassembly listing of the machine code generated. The -S, --asm, and --interleave compiler command-line options are disabled. Projects will not compile with Target -> Listing -> C Compiler Listing enabled.
- The compiler and assembler do not generate position-independent code or data. The --apcs /ropi /rwpi /pic/ pid compiler and assembler command line options are disabled.
- The assembler and linker create Symbolic Output Format objects which cannot be linked with third-party linker utilities. Fully licensed tools generate standard ELF/DWARF files which may be used with third-party utilities.

Pack Installation



List of Supported devices



NUCLEO-F401RE

ISEN ALL IS DIGITAL! | yncréa

Home › Evaluation Tools › Product Evaluation Tools › MCU Eval Tools › STM32 MCU Eval Tools › STM32 MCU Nucleo › **NUCLEO-F401RE**

NUCLEO-F401RE ACTIVE

STM32 Nucleo-64 development board with STM32F401RE MCU, supports Arduino and ST morpho connectivity

Download Databrief

QUICK VIEW | RESOURCES | TOOLS AND SOFTWARE | SAMPLE & BUY | QUALITY & RELIABILITY

The STM32 Nucleo board provides an affordable and flexible way for users to try out new concepts and build prototypes with the STM32 microcontroller, choosing from the various combinations of performance, power consumption and features. The Arduino™ Uno V3 connectivity support and the ST morpho headers allow to expand easily the functionality of the STM32 Nucleo open development platform

with a wide choice of specialized shields. The STM32 Nucleo board does not require any separate probe as it integrates the ST-LINK/V2-1 debugger and programmer. The STM32 Nucleo board comes with the STM32 comprehensive software HAL library together with various packaged software examples, as well as direct access to the ARM® mbed™ online resources at http://mbed.org.

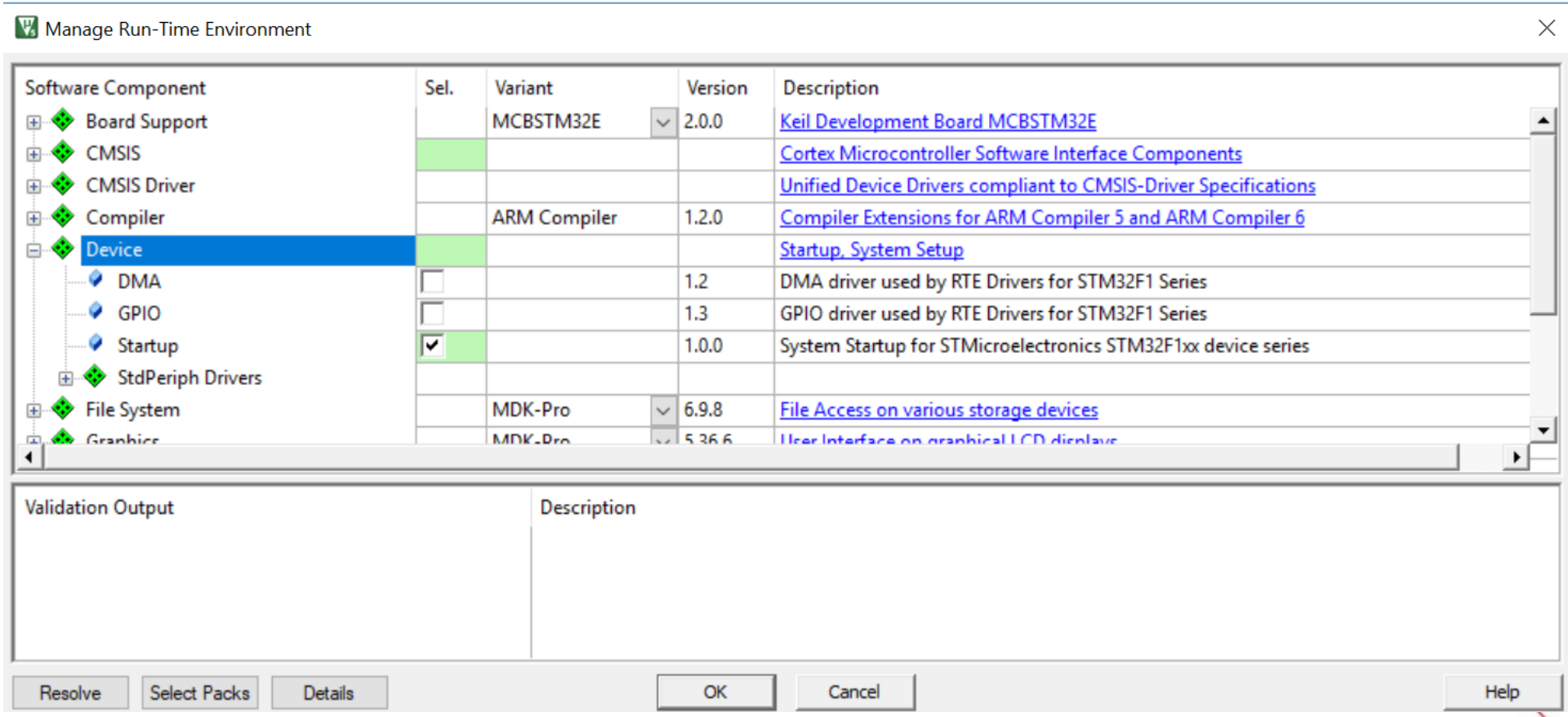
Startup file

The startup file provides the necessary code to initialize the C runtime envitonment and runs from the processor reset vector to the main() function in the application code.

It provides the processor vector table, stack, and variable initialization.

It does not provide any hardware initialization for user peripherals or the processor system peripherals.
Typically, the startup code resides in a file named **startup_*device_family*.s**

Startup file Creation



In order to create the start file, the box Startup must be checked

Information after a Build

```
Build target 'BitBand'
assembling STM32F10x.s...
compiling main.c...
linking...
Program Size: Code=448 RO-data=268 RW-data=4 ZI-data=612
".\BitBand.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed:  00:00:01
```

Name	Meaning
Code	Size of the Executable image
RO data	Size of the code constants in the flash memory
RW data	Size of the initialized variables in SRAM
ZI data	Size of the uninitialized variables in SRAM

ISEN | école d'ingénieurs
ALL IS DIGITAL!

CORTEX-M Tutorial

Cortex-M Architecture

Cortex-M Instruction Set

- ✓ Cortex-M processors are Reduced Instruction Set Computer (RISC)
  - Cortex-M0 has 56 instructions
  - Cortex-M3 has 74 instructions
  - Cortex-M4 has 137 instructions with an option of additional 32 instructions for the FPU

ARM Instructions

ISEN ALL IS DIGITAL! | yncréa

2 instruction sets

- ✓ ARM instruction set would allow code to be written for maximum performance as they are 32-bit instructions
- ✓ Thumb code would achieve a greater code density as it is 16-bit instructions
- ✓ Cortex-M processors are code compatible with the original Thumb instruction set, but they are designed to execute an extended version of the Thumb instruction set called Thumb-2.
- ✓ Thumb-2 is a blend of 16 and 32 bit instructions that has been designed to be very C friendly and efficient.

Cortex-M Memory Map and registers

ISEN ALL IS DIGITAL! | yncréa

4 GB Memory Map

Vendor Specific	
Private peripheral bus - external	
Private peripheral bus- internal	
External Device	1.0GB
External RAM	1.0GB
Peripheral	0.5GB
SRAM	0.5GB
Code	0.5GB

Registers

R0-R12 general purpose register
R13 (MSP/PSP Stack Pointer)
R14 (LR Link Register)
PC (Program Counter)
PSR (Program Status Register)
PRIMASK
FAULTMASK
BASEPRI
CONTROL

Memory Map

Figure 8. Memory map

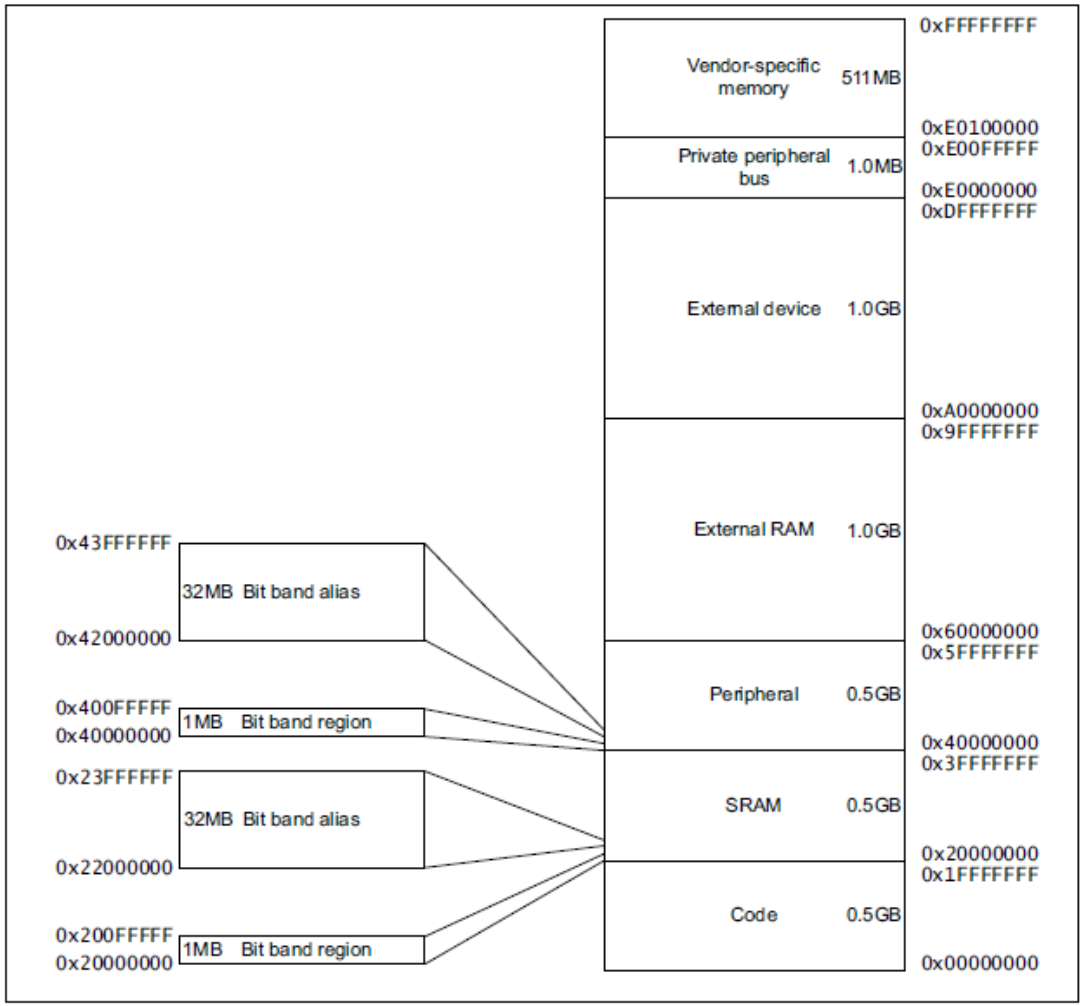


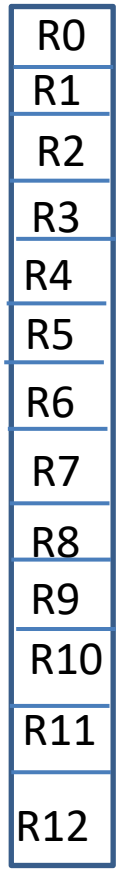
Table 12. Memory access behavior

Address range	Memory region	Memory type	XN	Description
0x00000000-0x1FFFFFFF	Code	Normal(1)	-	Executable region for program code. Can also put data here.
0x20000000-0x3FFFFFFF	SRAM	Normal(1)	-	Executable region for data. Can also put code here. This region includes bit band and bit band alias areas, see *Table 13 on page 31*.
0x40000000-0x5FFFFFFF	Peripheral	Device(1)	XN(1)	This region includes bit band and bit band alias areas, see *Table 14 on page 31*.
0x60000000-0x9FFFFFFF	External RAM	Normal(1)	-	Executable region for data.
0xA0000000-0xDFFFFFFF	External device	Device(1)	XN(1)	External Device memory
0xED000000-0xED0FFFFF	Private Peripheral Bus	Strongly-ordered(1)	XN(1)	This region includes the NVIC, system timer, and system control block.
0xED100000-0xFFFFFFFF	Memory mapped peripherals	Device(1)	XN(1)	This region includes all the STM32 standard peripherals.

Cortex-M Registers

- ✓ The Cortex-M processors apply a load and store method of operations
- ✓ This means that to do any kind of data processing is performed from registers
- ✓ Instructions such as ADD and SUBTRACT: data must first be loaded into the CPU registers. The data processing is then executed and the result is stored back in the main memory.

General Purpose Registers



- ✓ R0-R12 are 32-bit wide general purpose registers
- ✓ They can be accessed by all the Thumb-2 load and store instructions
- ✓ They are used by the compiler

R13 MSP/PSG Register

- ✓ R13 is used by the compiler as a stack pointer
- ✓ It is actually a banked register with two R13 registers
  - MSP: Main Stack Pointer
  - PSP: Process Stack Pointer
- ✓ When out of reset, the processor runs in « simple » mode with one stack pointer
- ✓ It is possible to enable the second R13 register by writing to the Cortex control register

R14 Register

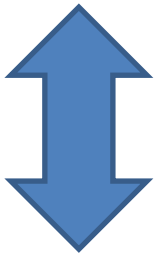
- ✓ R14 is called the link register
- ✓ When a procedure is called the return address is automatically stored in R14

PSR Register

- ✓ PSR stands for Program Status Register
- ✓ As its name implies, the PSR contains all the CPU status flags
- ✓ The PSR has three alias registers
  - Application Program Status Register (APSR)
  - Interrupt Program Status Register (IPSR)
  - Execution Program Status Register (EPSR)
- ✓ Each of these alias registers contains a subset of the full register flags and can be used as shortcut if it is necessary to access part of the PSR.
- ✓ The PSR is generally referred to as the xPSR to indicate the full register rather than any of the alias subsets.

PSR Register Contents

APSR	N	Z	C	V	Q										
IPSR									Exception Number						
EPSR						ICI/IT	T		ICI/IT						



PSR	N	Z	C	V	Q	ICI/IT	T		ICI/IT	Exception Number

- ✓ In a normal application program, the code does not make explicit access to the xPSR or any of it alias registers
- ✓ Any use of the xPSR is made by compiler generated code.
- ✓ But as a programmer it is mandatory to have an awareness of the xPSR and the flags contained in it.

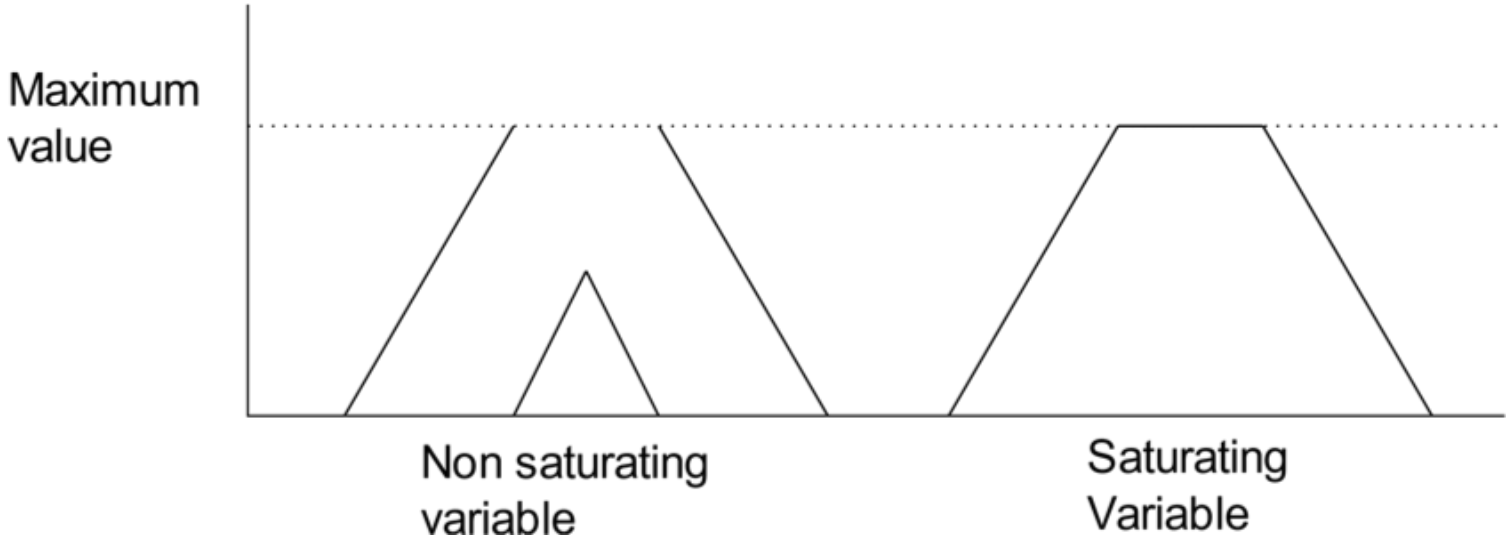
xPSR Register Contents

- ✓ The most significant four bits of the xPSR are the condition code bits
  - Negative
  - Zero
  - Carry
  - oVerflow
- ✓ These are set and cleared depending on the results of a data processing instruction.
  - SUB R8,R6 perform a subtraction and do not update the condition code flags
  - SUBS R8,R6 perform a subtraction and update the conditions code flags
- ✓ This allows the compiler to perform an instruction set that updates the condition code flags, then perform instructions that do not modify the flags and then perform a conditional branch on the state of the xPSR condition codes.

xPSR Register Q bit

- ✓ The Q bit is the saturation flag. The Cortex-M3/4 have a special set of instructions called the saturated math instructions
- ✓ If a normal variable reaches its maximum value and is incremented further, it will round to zero. Similarly, if a variable reaches its minimum value and is then decremented, it will round to the maximum value.
- ✓ It is a critical problem in motor control and safety applications.
- ✓ The CortexM3/4 saturated math instructions prevent this kind of « roll around »
- ✓ When saturated math is used, if the variable reaches its maximum or minimum value it sticks at that value.
- ✓ The « Q » bit is a sticky bit and must be cleared by the application code.

Use of Saturated Math Instructions



The standard math instructions are not used by the C compiler by default. To make use of the saturated math instructions, you have to access them by using compiler intrinsic or CMSIS-core functions

- ✓ uint32_t __SSAT(uint32_t value, uint32_t sat)
- ✓ Uint32_t __USAT(uint32_t value, uint32_t sat)

Interrupts and Multicyle Instructions

ISEN
ALL IS DIGITAL!
yncréa

- ✓ Most of the Cortex-M processor instructions are executed in a single cycle. However, some instructions such as load store multiple, multiple and divide take multiple cycles
- ✓ If an interrupt occurs while these instructions are executing, they have to be suspended while the interrupt is served
- ✓ Once the interrupt has been served, the multicycle instructions must be resumed
- ✓ The ICI field is managed by the Cortex-M processor, not by the application code
- ✓ The ICI field means when an exception is raised reaching the start of the interrupt routine will always take the same amount of cycles regardless of the instruction currently being executed by the CPU

Instruction Condition Cycles

Condition Code	xPSR Flags Tested	Meaning
EQ	Z=1	Equal
NE	Z=0	Not Equal
CS ou HS	C=1	Higher or same (unsigned)
CC ou LO	C=0	Lower (unsigned)
MI	N=1	Negative
PL	N=0	Positive or Zero
VS	V=1	Overflow
VC	V=0	No Overflow
HI	C=1 and Z=0	Higher (unsigned)
LS	C=0 and Z=1	Lower or same (unsigned)
GE	N=V	Greater than or equal(signed)
LT	N!=V	Less than(signed)
GT	Z=0 and N=V	Greater than (signed)
LE	Z=1 and N!=V	Less than or equal (signed)
AL	None	Always execute

Branching Issues and Tricks

- ✓ The 3-stage pipeline allows 3 operations in parallel: fetch, decode and execute
- ✓ Disadvantage: every time the processor makes a jump, the pipeline has to be flushed and refilled -> introduces a big hit on performance as the pipeline has to be refilled before the execution of instructions can resume
- ✓ The CortexM3/4 reduce the branch penalty by having an instruction fetch unit that can carry out speculative branch target fetches from the possible branch address so the execution of the branch targets can start earlier
- ✓ For small conditional branches, the Cortex-M processor has another trick with IF THEN blocks

IF THEN Blocks

For a small conditional branch like:

```
if (a==0x0123) {
        i++;
        } else {
        i--;}
```

Can compile to less than four instructions, the Cortex-M can compile as an IF THEN condition block. The instructions inside the IF THEN block are extended with a condition code. This condition code is compared to the state of the condition code flags in the PSR. If the condition matches the state of the flags, then the instruction will be executed and if it does not then the instruction will enter the pipeline but will be executed as a no operation(NOP).

This technique eliminates the branch and hence avoids the need to flush and refill the pipeline.

IF THEN Blocks

To trigger an IF THEN block, the data processing instructions are used to update the PSR condition codes. By default, most instructions do not update the condition codes unless they have an S suffix added to the assembler opcode. This gives the compiler a great deal of flexibility in applying the IF THEN condition

OpCode	comments
ADDS R1,R2,R3	Perform and add and set the xPSR flags
ADD R2, R4, R5	Do some other instructions but do not modify the xPSR
ADD R5, R6, R7	
IT VS	IF THEN block conditional on the first ADD instruction
SUBVS R3, R2, R4	

T bit

- ✓ The T bit is the Thumb bit
- ✓ This is a legacy bit from the earlier ARM CPUs and is set to one when the processor is released from reset
- ✓ If the code tries to clear this bit, it will be set to zero but a fault exception will also be raised
- ✓ In previous CPUs, The T bit was used to indicate that the Thumb 16-bit instruction set was running.
- ✓ It is included in the Cortex-M PSR to maintain compatibility with earlier ARM CPUs and allow legacy 16-bit thumb code to be executed on the Cortex-M processors.

Exception Number Field

- ✓ The final field in the PSR is the exception number field
- ✓ The Cortex NVIC can support up to 256 exceptions sources
- ✓ When an exception is being processed, the exception number is stored here
- ✓ This field is not used by the application code when handling an interrupt, but it can be a useful reference when debugging

BUSES

- ✓ While the cortex-M memory map is a linear 4GB address space with no paged region or complex addressing modes, the microcontroller memory and peripherals are connected to the Cortex-M processor by a number of different buses.
- ✓ The 1st 0.5GB of the address space is reserved for executable code and code constants. This region has two dedicated buses
  - The ICODE bus is used to fetch code instructions
  - The DCODE bus is done to fetch code constants
- ✓ The remaining user memory space (internal SRAM and peripherals, plus the external RAM and peripherals) are accessed by a separate system bus (AHB and APB bus)
- ✓ For a programmer, mainly the knowledge of ANH and APB bus is necessary

Code execution limited by Flash Memory

- ✓ Thumb-2 instructions are executed in a single cycle and the Cortex M3 can run up to 200MHz
- ✓ Current flash memory (2014) used to store the program has an access time of around 50MHz.
- ✓ Basic problem of pulling instructions out of the flash memory fast enough to feed the Cortex-M
- ✓ Various ways to overcome this issue but typically:
  - Flash memory will be arranged as 64- or 128-bit wide memory, so one read from the flash memory can load multiple instructions.
  - These instructions are then held in a « memory accelerator» unit which then feeds the instructions of the Cortex-M processor
  - The memory accelerator is a form of simple cache unit that is designed by the silicon vendor.
  - Normally, this unit is disabled after reset , so you need to enable it or the cortex-M processor will be running directly from the flash memory.
  - The overall performance of the Cortex-M depends on how successfully this unit has been implemented by the silicon vendor.

BUSES

- ✓ CortexM3-M4 contain a single entry data write buffer
- ✓ This allows the CPU to make an entry into the write buffer and continue on the next instruction while the write buffer complete the write to the real SRAM.
- ✓ If the write buffer is full, the CPU is forced to wait until it has finished its current write.
- ✓ It is normally transparent to the application code, but in some cases it is necessary to wait until the write has finished before continuing program execution.

ISEN | école d'ingénieurs
ALL IS DIGITAL!

CORTEX-M Tutorial

CMSIS

CMSIS Introduction

- ✓ CMSIS stands for : Cortex Microcontroller Software Interface Standard
- ✓ Standard defined by ARM
- ✓ Purpose: Allow easy integration of source code from multiple sources
  - Cortex-M based microcontrollers is the de facto industry standard for 32-bit microcontrollers
  - Explosive growth of devices, but same tools and skills regardless of the silicon vendor
- ✓ CMSIS is gaining increasing support through the industry and should be adopted for new projects.

CMSIS Specifications

- ✓ The main aim of CMSIS is to improve software portability and reusability across different microcontrollers and tool chains
- ✓ Once learned, CMSIS helps to speed up software and development through the use of standardized software functions
- ✓ CMSIS consists of five interlocking specifications that support all Cortex-M-based microcontrollers:
  - CMSIS Core
  - CMSIS RTOS
  - CMSIS DSP (Digital Signal Processing)
  - CMSIS SVD (System View Description)
  - CMSIS DAP (Debug Access Port)
- ✓ CMSIS is a simple framework that helps to integrate various source code
- ✓ Each of the CMSIS specifications are not complicated and can be learned easily.

What CMSIS is not

- ✓ CMSIS is not a complex abstraction layer that forces to use a complex and bulky library
- ✓ CMSIS does not attempt to degrade peripherals by providing standard profiles that make different manufacturer's peripherals work the same way
  - CMSIS core specification takes a very small amount of resources (about 1K of code and 4 bytes of RAM) and just standardizes the way Cortex-M microcontroller registers are accessed
- ✓ CMSIS does not affect the way the code is developed and does not force to adopt a particular methodology

Where to find CMSIS Specifications

ISEN ALL IS DIGITAL! | yncréa

- ✓ Can be read on http://arm-software.github.io/CMSIS_5/Core/html/index.html
- ✓ MDK-ARM: CMSIS specifications are integrated into the MDK-ARM toolchain and the CMSIS documentation is available from the online help

CMSIS Core

- ✓ The core specification provides a minimal set of functions and macros to access the key Cortex-M processor registers
- ✓ Defines a function to configure the microcontroller oscillators and clock tree in the startup (hence the device is ready for use when the main() function of the Code is reached).
- ✓ Standardizes the naming of the device peripheral registers
- ✓ Standardizes support for the instrumentation trace during debug sessions

CMSIS RTOS

- ✓ Provides a standard API for an RTOS (set of wrapper functions that translate the CMSIS RTOS API of the specific RTOS that the developer uses)
- ✓ Keil RTX RTOS was the first RTOS to support the CMSIS RTOS API and it has been released as an open source reference implementation
- ✓ RTX can be compiled with various compilers: GCC, IAR, Keil…
- ✓ It is licensed with a three-clause Berkley Software Distribution (BSD) license that allows its unrestricted use in commercial and noncommercial applications

CMSIS DSP: Digital Signal Processing

ISEN ALL IS DIGITAL! | yncréa

- ✓ Cortex-M4 is a « Digital Signal Controller » with a number of DSP functions
- ✓ Developing a real-time DSP system is not trivial
- ✓ To include DSP functions smoothly, CMSIS includes a DSP library that provides over 60 of the most commonly used DSP mathematical functions
- ✓ These functions are optimized to run on the cortex-M4 but can also be compiled to run on the Cortex-M3

CMSIS SVD: System View Description

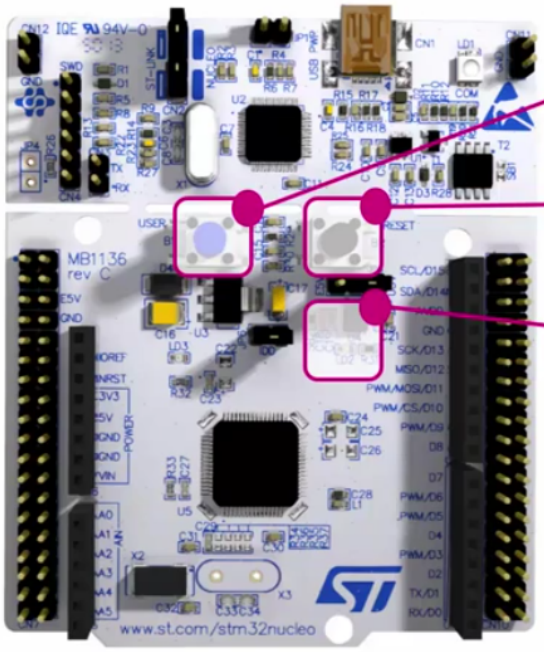
- ✓ The debugger must provide peripheral view windows that show the developer the current state of the microcontroller peripheral registers
- ✓ With the growth in the numbers of Cortex-M vendors and the rising number and complexity of on-chip peripherals, it is impossible for any given tools vendor to maintain
- ✓ To overcome this hurdle the CMSIS debug specification defines a « System Viewer Description » (SVD) file
- ✓ This file is provided and maintained by the silicon vendor and contains a complete description of the microcontroller peripheral registers in an XML format
- ✓ This file is then imported by the development tool, which uses it to automatically construct the peripheral debug windows for the microcontroller.

CMSIS DAP: (Debug Access Port)

- ✓ The CMSIS DAP specification defines the interace protocol for a hardware debug unit that sits between the host PC and the Debug Access Port (DAP) of the microcontroller
- ✓ Allow any software tool chain that support CMSIS DAP to connect to hardware debug unit that also supports CMSIS DAP

NUCLEO Board with STMF401

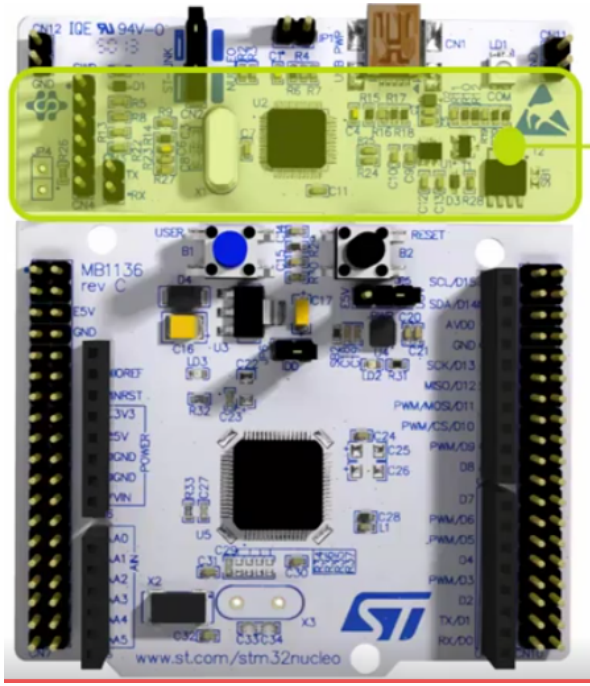
NUCLEO Board with STMF401



One user BUTTON connected to PC13

Reset BUTTON connected to NRST pin

Green LED connected to pin PA5

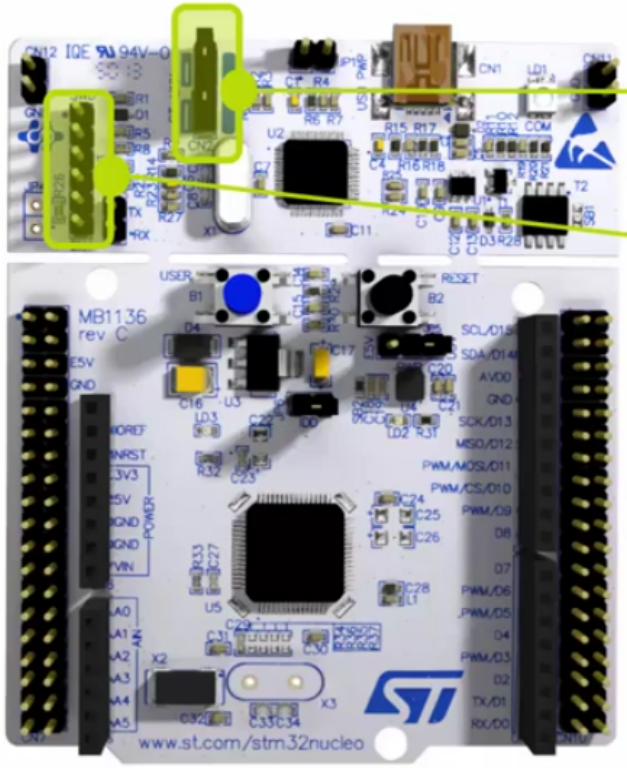


Integrated ST-Link/V2-1
Mass storage device flash programming

Mass storage programmer
(drag and drop binary programming)

Virtual com port bridge connected to
STM32F401RE pins

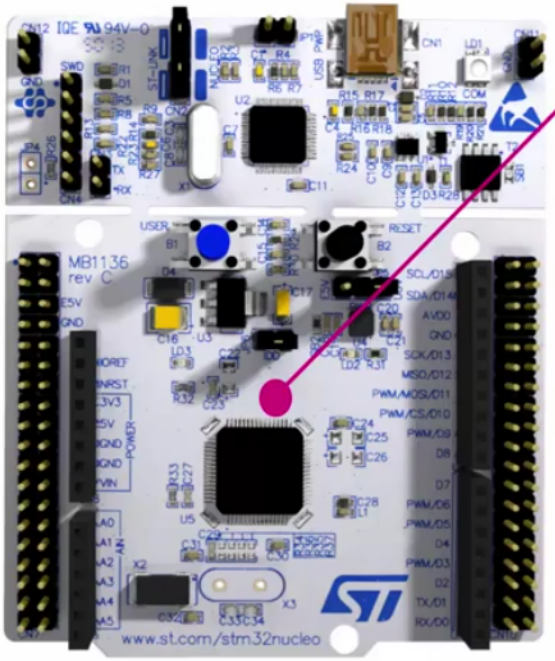
ST-Link V2-1 debugger connected to
STM32F401RE



By disconnecting the jumpers allow to debug/program other STM32 MCUs

Connector with SWD for STM32 connection

VDD - Application voltage (not used)
TCK – SWD clock
GND – ST-Link ground
TMS – SWD data
NRST – Application reset
SWO – Application SWO output



Schematic from STM32F401RE-NNUCLEO is possible to found in UM1724