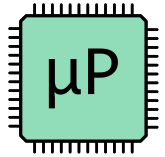
STM32 Datasheet and User Manual

Lecture 5

Microprocessor-based Systems (E155)

Prof. Josh Brake



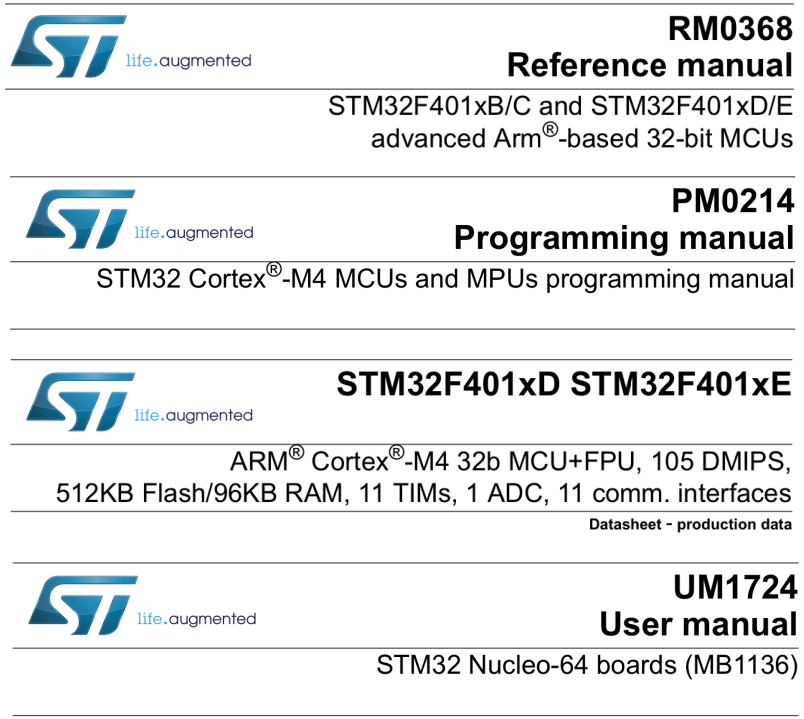
From last time…

- What's going on in startup.c?
  - Setup vector table
    - Stack pointer
    - reset_handler function location (`startup()` in our case)
  - Copy variables from flash to RAM
  - Setup initialization for global/extern variables in Block Started by Symbol section (.bss)
  - Jump to main
- Make sure that you do the startup before you do anything else! (hint: make sure you link your files correctly. You need to make sure your vector table gets mapped to `0x00` and that `startup()` executes before `main()`).

Outline

Documentation for NucleoF401RE board

- STM32F401RE
  - Datasheet
  - User Manual
  - Programming Manual (All STM32 Cortex-M4)
- Nucleo-64 User Manual


life.augmented

RM0368
Reference manual

STM32F401xB/C and STM32F401xD/E
advanced Arm®-based 32-bit MCUs

life.augmented

PM0214
Programming manual

STM32 Cortex®-M4 MCUs and MPUs programming manual

life.augmented

STM32F401xD STM32F401xE

ARM® Cortex®-M4 32b MCU+FPU, 105 DMIPS,
512KB Flash/96KB RAM, 11 TIMs, 1 ADC, 11 comm. interfaces

Datasheet - production data

life.augmented

UM1724
User manual

STM32 Nucleo-64 boards (MB1136)

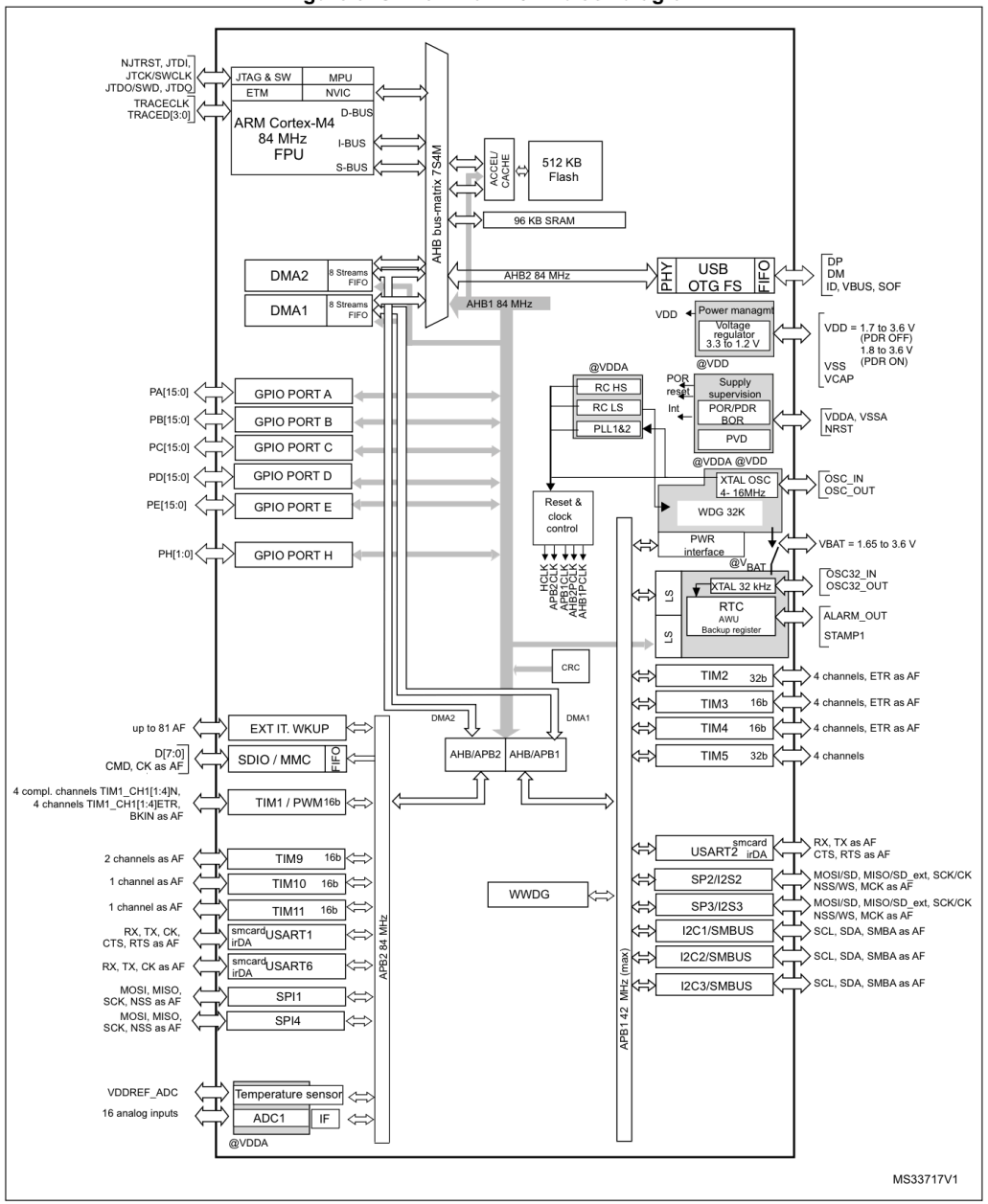
What is each piece of documentation for?

- Datasheet – what is it?
  - Functional overview
  - Pinouts
  - Memory mapping
  - Electrical characteristics
  - Package details
- Reference manual – how do I use it?
  - How to use memory and peripherals
    - Register mappings
- Programming manual – how do I program it?
  - Cortex-M4 processor details
  - Instruction set
  - Core peripherals

The STM32F401RE

- Cortex-M4 Microprocessor
- Up to 84 MHz
- UART/USARTs, GPIO, Timers, I2C, SPI, RTC

Figure 3. STM32F401xD/xE block diagram



p.14 STM32F401RE Datasheet

Part Numbering

Table 86. Ordering information scheme

Example: STM32 F 401 C E Y 6 TR

Device family

STM32 = ARM-based 32-bit microcontroller

Product type

F = General-purpose

Device subfamily

401 = 401 family

Pin count

C = 48/49 pins

R = 64 pins

V = 100 pins

Flash memory size

D = 384 Kbytes of Flash memory

E = 512 Kbytes of Flash memory

Package

H = UFBGA

T = LQFP

U = UFQFPN

Y = WLCSP

Temperature range

6 = Industrial temperature range, –40 to 85 °C

Packing

TR = tape and reel

No character = tray or tube

Activity: Datasheet hunting

- Don’t use search
- Look for the value and click yes button on Zoom when you think you’ve found it
- Give answer and reference (place where you found it)

Questions!

- Memory address for Timer 1
- What is flash memory base address?
- How much flash memory does the board have?
- How much RAM?
- Max clock speed?
- How many GPIO pins?
- What package is the STM32F401RE on our board?

Questions!

- Input output logic voltage levels?
  - VOH
  - VOL
  - VIH
  - VIL
- Max operating temperature?
- Max operating voltage?
- Max current draw for GPIO pin?

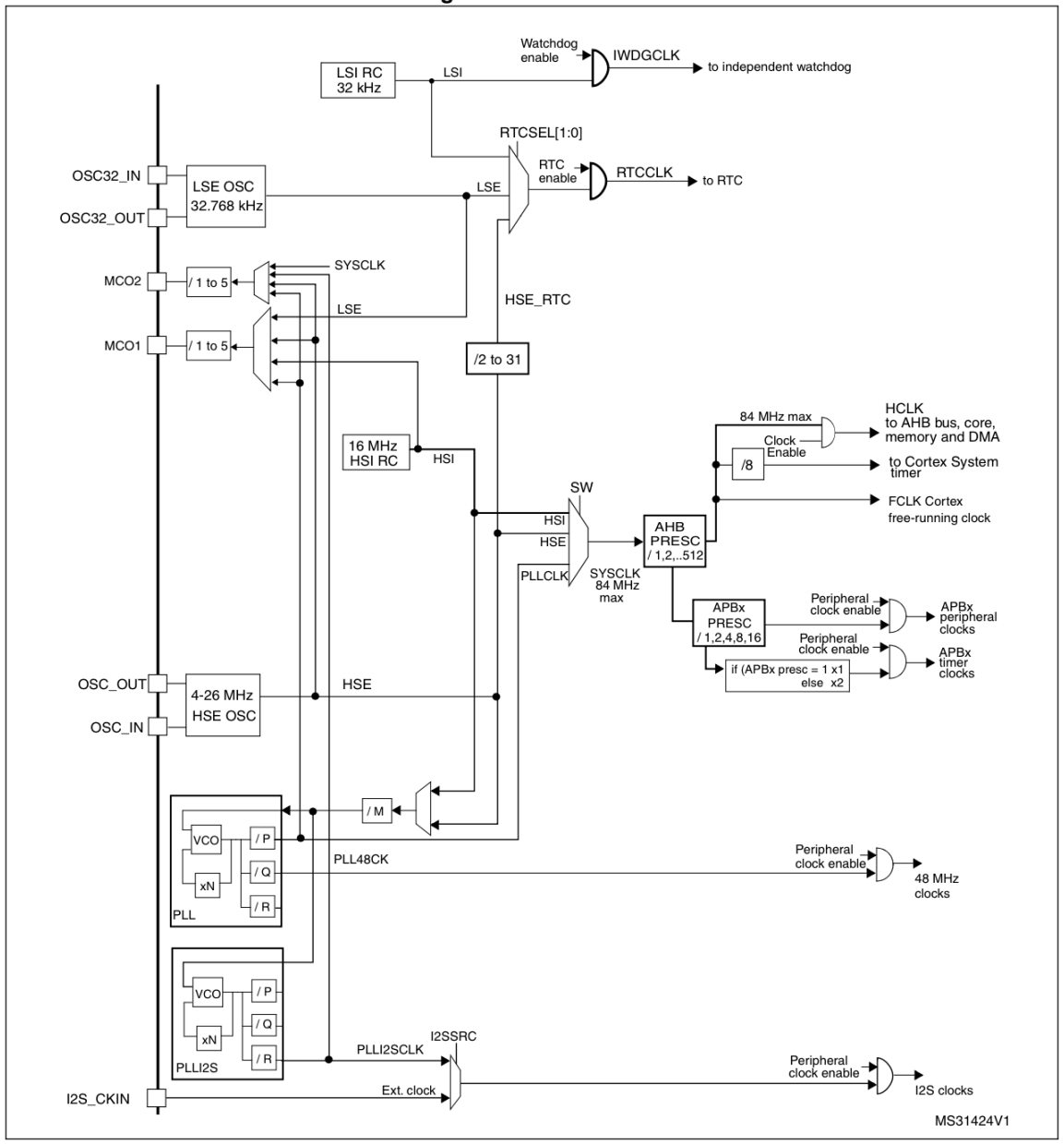
Questions!

- What is the maximum voltage I could apply to a pin before destroying it?
- What is the maximum voltage I could apply to while expecting proper operation?
- Which pin is connected to the OSC_IN input?

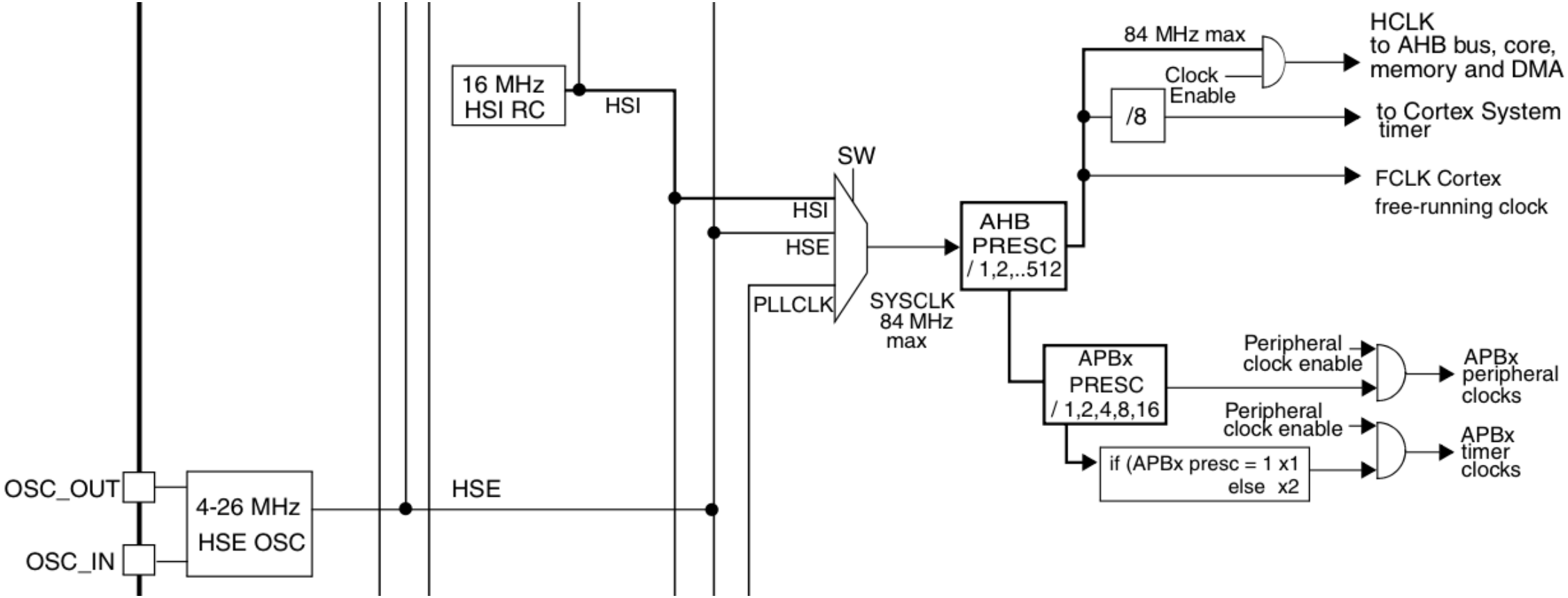
Clock configuration

- What clock source is used at boot?
- Do we have an external clock source available on our Nucleo board?
  - What frequency is it?
  - How do we configure it?

Figure 12. Clock tree



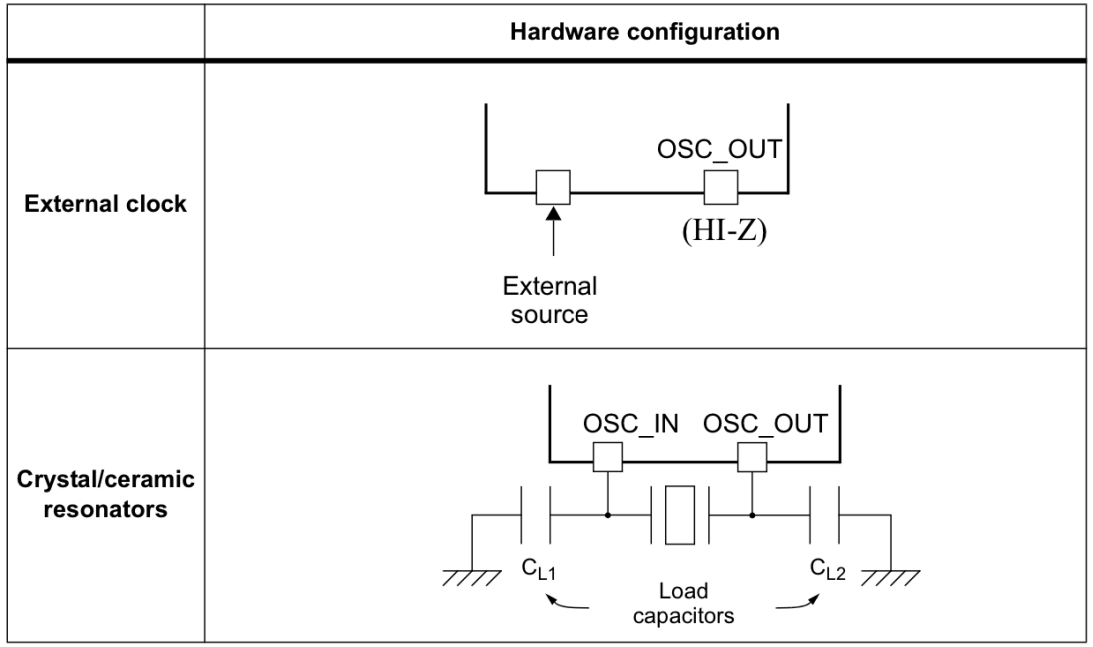
AHB/APB Clock Tree Inputs and Outputs



HSE Config

- Set HSEBYP and HSEON in RCC_CR
- External signal drives OSC_IN with ~50% duty cycle while OSC_OUT should be left HI-Z.

Figure 13. HSE/ LSE clock sources



External source (HSE bypass)

In this mode, an external clock source must be provided. You select this mode by setting the HSEBYP and HSEON bits in the *RCC clock control register (RCC_CR)*. The external clock signal (square, sinus or triangle) with ~50% duty cycle has to drive the OSC_IN pin while the OSC_OUT pin should be left HI-Z. See *Figure 13*.

External crystal/ceramic resonator (HSE crystal)

The HSE has the advantage of producing a very accurate rate on the main clock.

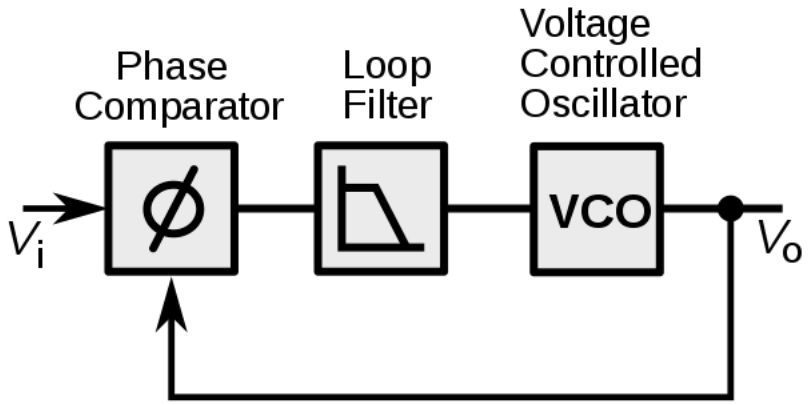
The associated hardware configuration is shown in *Figure 13*. Refer to the electrical characteristics section of the *datasheet* for more details.

The HSERDY flag in the *RCC clock control register (RCC_CR)* indicates if the high-speed external oscillator is stable or not. At startup, the clock is not released until this bit is set by hardware. An interrupt can be generated if enabled in the *RCC clock interrupt register (RCC_CIR)*.

The HSE Crystal can be switched on and off using the HSEON bit in the *RCC clock control register (RCC_CR)*.

What if we want to get to the fastest clock speed available?

- PLL generates higher frequency signal from lower frequency input.
- Does this using a feedback loop.



6.3.2 RCC PLL configuration register (RCC_PLLCFGR)

Address offset: 0x04

Reset value: 0x2400 3010

Access: no wait state, word, half-word and byte access.

This register is used to configure the PLL clock outputs according to the formulas:

- $f_{(VCO\ clock)} = f_{(PLL\ clock\ input)} \times (PLLN / PLLM)$
- $f_{(PLL\ general\ clock\ output)} = f_{(VCO\ clock)} / PLLP$
- $f_{(USB\ OTG\ FS,\ SDIO,\ RNG\ clock\ output)} = f_{(VCO\ clock)} / PLLQ$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				PLLQ3	PLLQ2	PLLQ1	PLLQ0	Reserved	PLLSRC	Reserved				PLLP1	PLLP0
				rw	rw	rw	rw		rw					rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	PLLN									PLLM5	PLLM4	PLLM3	PLLM2	PLLM1	PLLM0
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

- PLL VCO output determined by PLLN/PLLM
- General output divided again by PLLP
- Special peripheral output divided by PLLQ

Phase locked loop (PLL)

6.2.3 PLL configuration

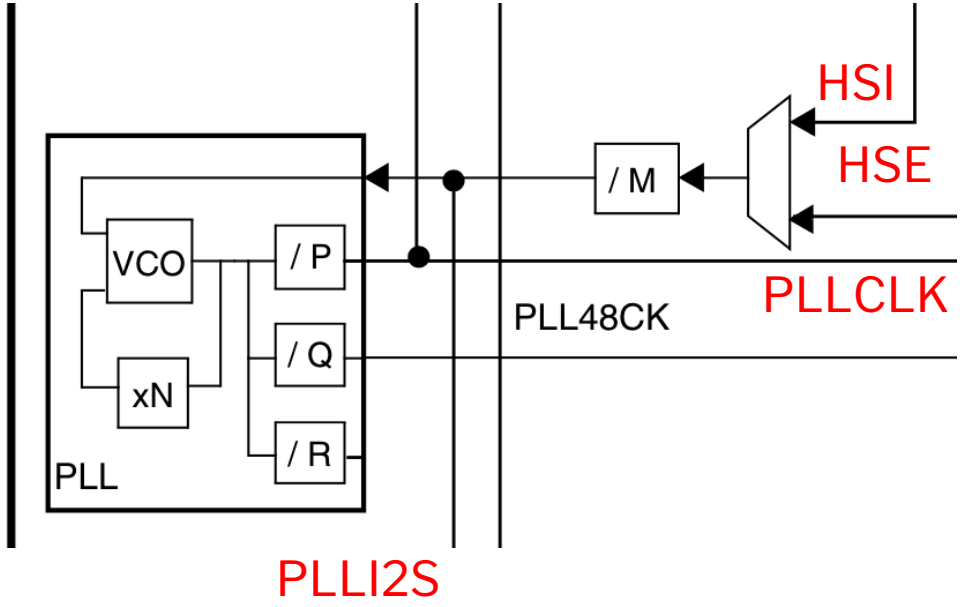
The STM32F401xB/C and STM32F401xD/E devices feature two PLLs:

- A main PLL (PLL) clocked by the HSE or HSI oscillator and featuring two different output clocks:
  - The first output is used to generate the high speed system clock (up to 84 MHz)
  - The second output is used to generate the clock for the USB OTG FS (48 MHz), the random analog generator (≤48 MHz) and the SDIO (≤48 MHz).
- A dedicated PLL (PLLI2S) used to generate an accurate clock to achieve high-quality audio performance on the I2S interface.

Since the main-PLL configuration parameters cannot be changed once PLL is enabled, it is recommended to configure PLL before enabling it (selection of the HSI or HSE oscillator as PLL clock source, and configuration of division factors M, P, Q and multiplication factor N).

Configuring the PLL

- Configure input clock (HSI/HSE)
- Set main divisor M
- Set multiplier N
- Set divisor P
- Turn on PLL



STM32F401RE RM p.94, Figure 12. Clock tree

6.3.2 RCC PLL configuration register (RCC_PLLCFGR)

Address offset: 0x04

Reset value: 0x2400 3010

Access: no wait state, word, half-word and byte access.

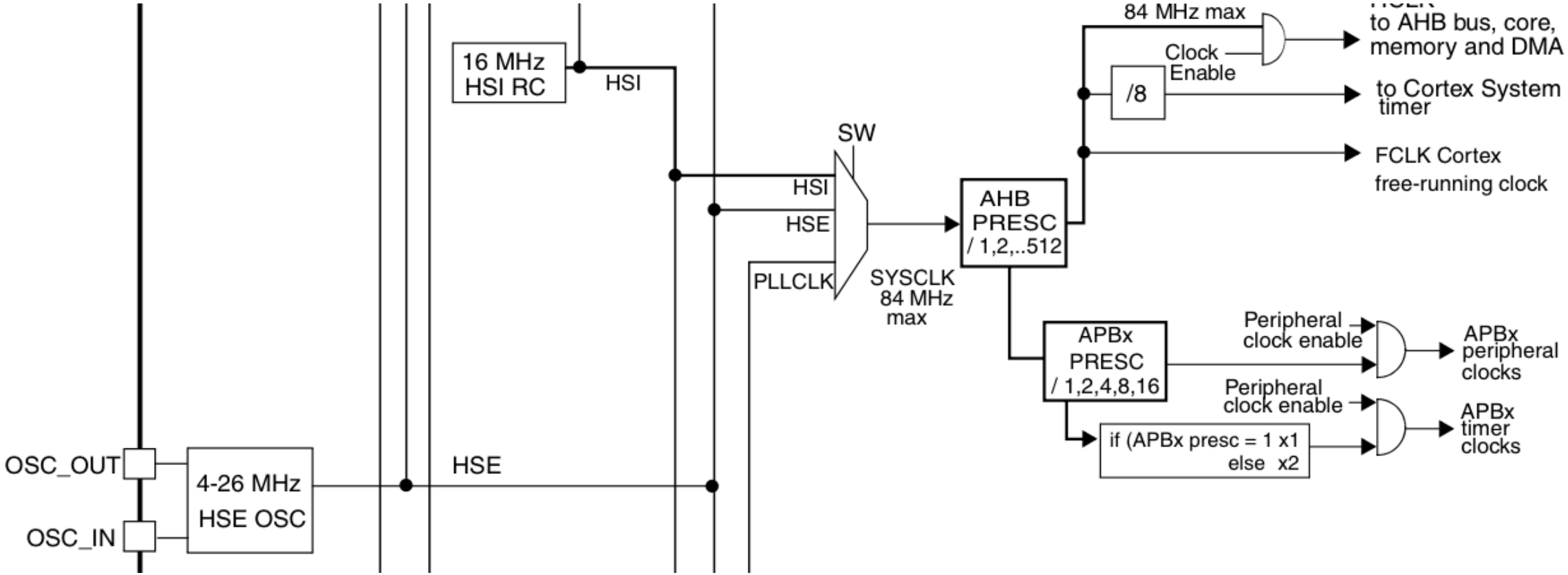
This register is used to configure the PLL clock outputs according to the formulas:

- $f_{(VCO\ clock)} = f_{(PLL\ clock\ input)} \times (PLLN / PLLM)$
- $f_{(PLL\ general\ clock\ output)} = f_{(VCO\ clock)} / PLLP$
- $f_{(USB\ OTG\ FS,\ SDIO,\ RNG\ clock\ output)} = f_{(VCO\ clock)} / PLLQ$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				PLLQ3	PLLQ2	PLLQ1	PLLQ0	Reserved	PLLSRC	Reserved				PLLP1	PLLP0
				rw	rw	rw	rw		rw					rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	PLLN									PLLM5	PLLM4	PLLM3	PLLM2	PLLM1	PLLM0
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

AHB/APB prescaling



STM32F401RE RM p.94, Figure 12. Clock tree

Bitfield structures

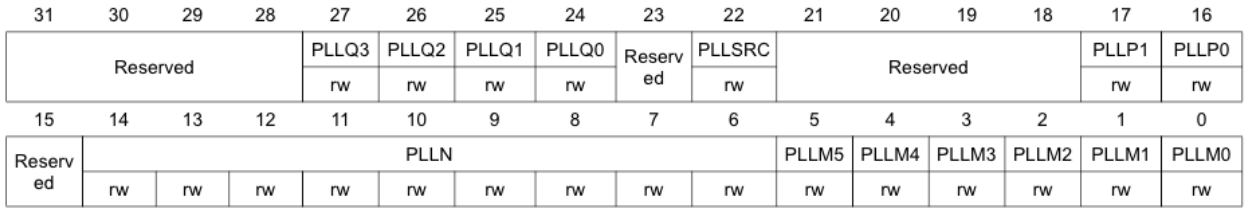
- A convenient way to interact with individual registers.

```
typedef struct {
volatile uint32_t <name1>  : width_in_bits;
volatile uint32_t <name2>  : width_in_bits;
volatile uint32_t <name3>  : width_in_bits;
...
} <structure_name>;
```

Bitfield structures: RCC Example

```
typedef struct {
volatile uint32_t PLLM     : 6;
volatile uint32_t PLLN     : 9;
volatile uint32_t          : 1;
volatile uint32_t PLLP     : 2;
volatile uint32_t          : 4;
volatile uint32_t PLLSRC   : 1;
volatile uint32_t          : 1;
volatile uint32_t PLLQ     : 4;
volatile uint16_t          : 4;
} RCC_PLLCFGR_bits;
```

```
typedef struct {
volatile uint32_t           RCC_CR;
volatile RCC_PLLCFGR_bits   RCC_PLLCFGR;
volatile uint32_t           RCC_CFGR;
...
} RCC;
```



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				PLLQ3	PLLQ2	PLLQ1	PLLQ0	Reserved	PLLSRC	Reserved				PLLP1	PLLP0
				rw	rw	rw	rw		rw					rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	PLLN									PLLM5	PLLM4	PLLM3	PLLM2	PLLM1	PLLM0
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

p. 105 STM32F401RE User Manual

```
#define RCC_BASE (0x40023800UL)          // base address of RCC
#define RCC_PTR ((RCC *) RCC_BASE)       // Pointer to RCC block of memory

RCC_PTR->RCC_PLLCFGR.PLLM = 10;          // Set 6-bit-wide PLLM field to 10
```

Summary

- Three main sources of information for STM32F401RE: datasheet, reference manual, programming manual
- Also can reference Nucleo-64 user manual for information on the specific Nucleo board
- General peripheral control (e.g., clock tree)
  - Look at block diagram
  - Scan registers
  - Find registers and specific bits that need to be set
  - Write code (structures, bitfield structures, etc.)

Lecture Feedback

- What is the most important thing you learned in class today?
- What point was most unclear from lecture today?