FAQ0059
Frequently Asked Questions

Clock enable process in the at32f403_clock.c

Questions:

Clock enable process in the at32f403_clock.c

Answer:

In the at32f403_clock.c, the system_clock_config() function is used to configure system clock. When the HEXT is directly used as the main system clock, or when HEXT is indirectly used as the clock source of PLL and then the PLL is used as the main system clock, the following code will be executed:

```
#define HEXT_STABLE_DELAY                    (5000u)
#define PLL_STABLE_DELAY                     (500u)
/* reset crm */
crm_reset();
crm_clock_source_enable(CRM_CLOCK_SOURCE_HEXT, TRUE);
/* wait for hext stable ,specially for AT32F403*/
wait_stbl(HEXT_STABLE_DELAY);
/* wait till hext is ready */
while(crm_hext_stable_wait() == ERROR)
{
}

/* config pll clock resource */
crm_pll_config(CRM_PLL_SOURCE_HEXT_DIV, CRM_PLL_MULT_48, CRM_PLL_OUTPUT_RANGE_GT72MHZ);
/* enable pll */
crm_clock_source_enable(CRM_CLOCK_SOURCE_PLL, TRUE);
/* wait till pll is ready */
while(crm_flag_get(CRM_PLL_STABLE_FLAG) != SET)
{
}
/* config apb2clk */
crm_apb2_div_set(CRM_APB2_DIV_2);
/* config apb1clk */
crm_apb1_div_set(CRM_APB1_DIV_2);
/* 1step: config ahbclk div8 */
crm_ahb_div_set(CRM_AHB_DIV_8);
/* select pll as system clock source */
crm_sysclk_switch(CRM_SCLK_PLL);
/* wait till pll is used as system clock source */
while(crm_sysclk_switch_status_get() != CRM_SCLK_PLL)
{
}
/* delay */
```

```
    wait_stbl(PLL_STABLE_DELAY);
    /* 3step: config ahbclk to target div */
    crm_ahb_div_set(CRM_AHB_DIV_1);
    /* update system_core_clock global variable */
system_core_clock_update();
```

In this code there are two parts marked in red font. Their definitions are as follows:

Red segment 1:
"wait_stbl(HEXT_STABLE_DELAY);" is a delay command. The reason for this delay command is that the flag ready flag is set too early, meaning that the crystal is not yet stabilized, and thus setting PLL or switching system clock during this period may cause system failure. HEXT_STABLE_DELAY is set as 5000, and delay time is around 2 ms. Based on actual measurements, if the HEXT is given with proper configuration, it is possible to start further operations 2ms after the ready flag bit is set, without causing system error.

Note: It is important to prevent this delay code from being removed by compiler when in use.

Red segment 2:

This section demonstrates step-by-step code procedure to set AHB bus clock frequency division. It is operated as follows: configure the AHB with a larger divider value before switching system clock to PLL (PLLCLK > 168 MHz), and then adjust the divider value step by step to the desired targeter value after the completion of system clock switching. The reason for this is that setting too large AHB bus frequency at the moment when the system clock is switched to high-frequency clock may trigger system failure.

crm_hext_stable_wait() is very relevant to HEXT vibration timing, as shown below:

```
    #define HEXT_STARTUP_TIMEOUT        ((uint16_t)0x3000) /*!< time out for hext start up */
    error_status crm_hext_stable_wait(void)
    {
      uint32_t stable_cnt = 0;
      error_status status = ERROR;
      while((crm_flag_get(CRM_HEXT_STABLE_FLAG) != SET) && (stable_cnt < HEXT_STARTUP_TIMEOUT))
      {
        stable_cnt ++;
      }
      if(crm_flag_get(CRM_HEXT_STABLE_FLAG) != SET)
      {
        status = ERROR;
      }
      else
      {
        status = SUCCESS;
      }
      return status;
}
```

After HEXT is enabled, code enters loop wait until the HEXT ready flag is set before moving to the next. But time counter remains counting during loop wait period, so that if HEXT ready flag bit is not set and the timeout defined in the HEXT_STARTUP_TIMEOUT is reached, HEXT start error occurs, forcing code to return to ERROR, which has to handled by users in the system_clock_config().

Based on actual measurements, a high-quality HEXT, along with proper hardware configuration, can be stabilized within around 800 μs, even a less-than-best HEXT with not-so-good hardware configuration can get stable within 10 ms. The HEXT_STARTUP_TIMEOUT is set to 0x3000, meaning that the HEXT takes around 20 ms to be stable under best compiling configuration. If the HEXT is enabled and its ready flag bit is not set in

20ms, pointing to the potential issues such as matching, solder void or damage on the HEXT. This is why the HEXT_STARTUP_TIMEOUT is provided with a reasonable timeout detect value, which can not only meet the duration required for HEXT stabilization but also help detect possible hardware problems.

Additionally, it is necessary to ensure that hardware circuit design conforms with related specifications. For passive crystal, there is a need to check if it is with spec. if not, it is up to the users to adjust the capacitance to meet the specifications. Further information, refer to *AN0078_AT32_MCU_HW_EMC_EFT* and appropriate datasheet through the official website of ARTERY TECH.

Type: MCU applications

Applicable products: ATF32F403

Main function: HEXT

Minor function: None

Document revision history

Date	Revision	Changes
2022.3.10	2.0.0	Initial release
2022.9.20	2.0.1	Added *Red segment 2*