Preface

The series of PY32T020 microcontrollers features a high-performance 32-bit ARM® Cortex®-M0+ core, a wide voltage range MCU. They are embedded with up to 32 Kbytes of flash memory and 4 Kbytes of SRAM memory, operating at a maximum frequency of 48 MHz. A variety of products with different packaging types are included.

This application note will help you understand the attentions for using the PY32T020 modules and start quick development.

Table 1 Applicable Products

Type	Product Series
Microcontroller Series	PY32T020

Contents

1. LSE Configuration

- First configure LSE_DRIVER and then enable LSE (LSEON=1).

2. SPI Configuration

SPI Mode	Receive/Transmit mode	SPI Fastest Speed
Slave Full-Duplex	Receive	PCLK/16
Slave Full-Duplex	Transmit	PCLK/16
Master Full-Duple	Receive	PCLK/2
Master Full-Duple	Transmit	PCLK/2

- When SPI sends 8-bit data, the SPI register must be cast to a uint8_t type: (*(__IO uint8_t *)&SPI->DR);
- In SPI slave transmit mode, after writing a value to the SPI DR register, a subsequent write cannot overwrite the previous value before it is sent. To overwrite, first reset the entire SPI module (using SPI1RST in RCC_APBRSTR2), then rewrite the DR value;
- In SPI half-duplex master receive mode with CPHA = 0, CPOL = 1, and a prescaler of 256, an extra clock cycle occurs;
- The SPI->SR.BSY bit clears during the last clock cycle of SPI communication. In polling mode, ensure the previous frame transmission completes before updating data for the next frame.

3. PWR Configuration

- The watchdog must be enabled when the MCU enters STOP mode;
- Use the hardware watchdog instead of the software watchdog to improve program reliability;
- Once enabled, the watchdog cannot be disabled by software. Use the RTC timer to wake up and feed the watchdog in low-power modes.

4. GPIO Configuration

- When a GPIO is used for touch sensing, do not simultaneously configure it for general-purpose output or alternate function modes;

- Initialize GPIO structures by assigning 0 to avoid undefined initial values;
- Ensure no GPIO pin experiences a negative voltage exceeding -0.3V.

5. IO Backflow Current Driving MCU Operation

5.1 Attentions

- Software configuration can prevent backflow current on IOs from powering the MCU when VCC is off.

5.2 Operation Procedure

- Hardware**:** Place 100Ω–1KΩ resistors in series with the corresponding IOs.
- Software**:** Configure the corresponding IOs as open-drain outputs before power-on initialization.
- Delay for 5 ms.
- Proceed with normal program initialization.

5.3 Code Example

```
int main(void)
{
    LL_IOP_GRP1_EnableClock(LL_IOP_GRP1_PERIPH_GPIOA);    /*Enable the GPIOA clock*/
    /*Configure pin PA1 as open-drain output */
    LL_GPIO_SetPinMode(GPIOA,  LL_GPIO_PIN_1,  LL_GPIO_OUTPUT_OPENDRAIN);
    LL_mDelay(5);                                                    /*Delay for 5ms*/
}
```

6. STOP mode operation precautions

- For Revision A chips, disable BOR before entering STOP mode. (Fixed in version B)
- For Revision A chips requiring BOR, enable the watchdog before STOP mode and feed it via periodic wake-ups. (Fixed in version B)

7. TK Configuration

- For applications with significant power noise or requiring EFT certification, avoid using PF5 for touch keys (TK).
- In PCB layout, maintain a distance of ≥4cm between switching power inductors and spring-loaded keys.

8. LED Constant-Current Output Configuration

- To use PB2: First enable PA0's constant-current function.
- To use PB3: First enable PA1's constant-current function.

9. I2C Slave Configuration

9.1 Operation Procedure

Upon receiving a STOP signal, perform:

- Disable the I2C module;
- Re-enable the I2C module;
- Enable ACK.

9.2 Code Example

```
LL_I2C_Disable(I2C1);
LL_I2C_Enable(I2C1);
LL_I2C_AcknowledgeNextData(I2C1, LL_I2C_ACK);
```

10. NRST Reset Precautions

- Ensure the active-low reset pulse width on the NRST pin exceeds 300μs.

11. ADC Power-On Calibration

11.1 Attentions

- Recalibrate the ADC when operating conditions change (VCC variation is the primary factor affecting offset; temperature is secondary).
- Perform software calibration before first use of the ADC module.

11.2 Operation Process

- Enable the ADC clock, ADCEN=1.
- Initialize the ADC.
- Calibrate the ADC.

11.3 Code Example

```
static void APP_AdcConfig()
{
  LL_APB1_GRP2_EnableClock(LL_APB1_GRP2_PERIPH_ADC1);          // Enable the ADC1
clock
  if (LL_ADC_IsEnabled(ADC1) == 0)
  {
    LL_ADC_StartCalibration(ADC1);                                    //Enable Calibration
#if (USE_TIMEOUT == 1)
    Timeout = ADC_CALIBRATION_TIMEOUT_MS;
#endif
while (LL_ADC_IsCalibrationOnGoing(ADC1) != 0)
    {
#if (USE_TIMEOUT == 1)                                  // Check if Calibration has Timed Out
      if (LL_SYSTICK_IsActiveCounterFlag())
      {
        if(Timeout-- == 0)
        {
        }
      }
#endif
    }
  LL_mDelay(1);
  }
}
```

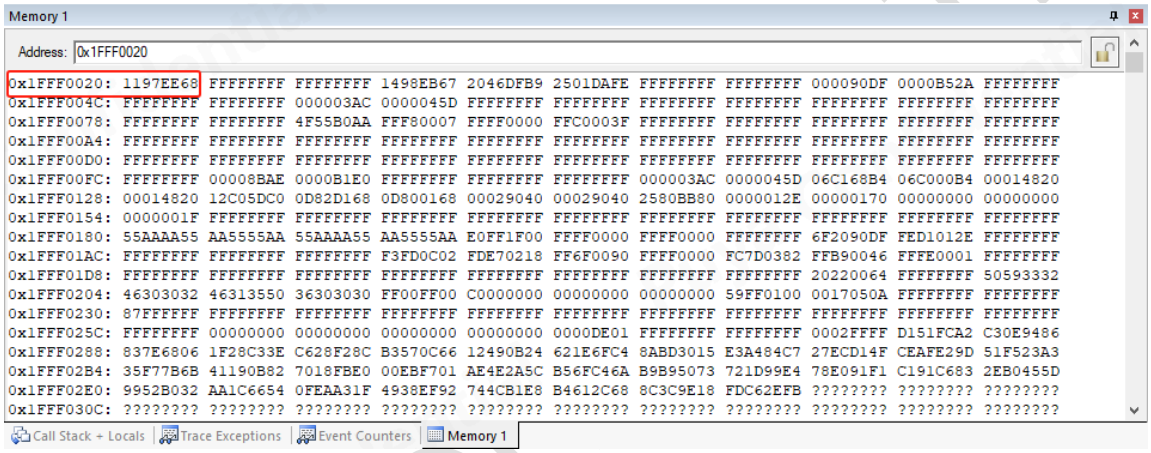
12. ADC Configuration

12.1 ADC Software Configuration

- Driving high-power devices directly from GPIOs can affect ADC sampling (e.g., 7-segment displays). Avoid ADC sampling during display updates, or add 10-100Ω series resistors to each display data line (adjust as needed).

12.2 Vreferint 1.2V

- The actual value of the internal 1.2V reference voltage (Vreferint) is stored in the Flash information area at address 0x1FFF0020. The high 16 bits contain the actual value; the low 16 bits contain its inverse. The read procedure is in Appendix 1.



13. Flash Configuration

- Flash supports page erase and page write operations only. Page size is 128 bytes. Start addresses must be page-aligned (e.g., 0x08000000, 0x08000080).
- Perform a page erase before each page write.

14. RCC Configuration

- After resetting a module on the APB bus with a high clock division factor, do not immediately read/write module registers. Insert __NOP() instructions exceeding the APB division factor (e.g., >10 NOPs for APB/8)

15. Version History

Version	Date	Update Records
V1.0	2024.3.20	Initial release
V1.1	2024.5.11	Add GPIO Configuration.
V1.2	2025.04.17	Add STOP, TK, LED, I2C chapter contents
V1.3	2025.06.04	Add NRST reset notes
V1.4	2025.06.23	Add ADC, IWDG, Flash, RCC, SPI, GPIO chapter contents
V1.5	2025.07.10	1．Merge the contents of SPI chapter 2．Add the contents of STOP chapter

Puya Semiconductor Co., Ltd.

Appendix 1

1.PY32T020 reads the Vreferint 1.2V actual value stored in the information area (see 12.2 for specific address).

```
#define HAL_VREF_INT                    (*(uint8_t *)(0x1fff3023))
#define HAL_VREF_DEC                    (*(uint8_t *)(0x1fff3022))
#define vref_int        (*(uint8_t *)(HAL_VREF_INT))          //Store the integer part of the reference voltage
#define vref_dec        (*(uint8_t *)(HAL_VREF_DEC))          // Store the fractional part of the reference voltage
float vref;                 //Reference voltage value

static uint8_t Bcd2ToByte(uint8_t Value)
{
    uint32_t tmp = 0U;
    tmp = ((uint8_t)(Value & (uint8_t)0xF0) >> (uint8_t)0x4) * 10U;
    return (tmp + (Value & (uint8_t)0x0F));
}

float read_1_2V(void)
{
    uint8_t data_vref_int,data_vref_dec;
    data_vref_int = Bcd2ToByte(HAL_VREF_INT);
    data_vref_dec = Bcd2ToByte(HAL_VREF_DEC);


  //Initialize all peripherals, flash interface， systick
    vref = data_vref_int/10;        //Calculate the reference voltage
    vref = vref + ((data_vref_int%10)*0.1 + data_vref_dec*0.001);
    return vref;
}
```