

PY32F030/PY32F003/PY32F002A 的 ADC 应用 注意事项

前言

PY32F030/PY32F003/PY32F002A 微控制器具有 1 个 12 位的 SARADC (successive approximation analog-to-digital converter)。该模块共有 12⁽¹⁾个要被测量的通道, 包括 10⁽¹⁾个外部通道和 2 个内部通道。支持多种使用方式。

本应用笔记将帮助用户了解 PY32F030/PY32003/PY32F002A 的 ADC 模块应用的注意事项并快速着手开发。

表 1. 适用产品

类型	产品系列
微型控制器系列	PY32F030、PY32F003、PY32F002A

1. PY32F030 的通道最大为 12 个, 外部通道最大为 10 个, PY32F003/PY32F002A 的通道最大为 11 个, 外部通道最大为 9 个。

目录

1	ADC 禁能	4
1.1	注意事项	4
1.2	操作流程	4
1.3	代码示例	4
2	ADC 复位	5
2.1	注意事项	5
2.2	操作流程	5
2.3	代码示例	5
3	ADC 校准	6
3.1	注意事项	6
3.2	操作流程	6
3.3	代码示例	6
4	ADC 仅使用通道 0 的情况	7
4.1	注意事项	7
4.2	操作流程	7
4.3	代码示例	7
5	ADC 连续转换模式	9
5.1	注意事项	9
5.2	操作流程	9
5.3	代码示例	9
6	ADC 单次模式	10
6.1	注意事项	10
6.2	操作流程	10
6.3	代码示例	10
7	ADC 如何利用 VREFINT (1.2V) 测量目标电压值	12
7.1	注意事项	12
7.2	操作流程	12
7.3	代码示例	12
8	AD 口配置注意事项	13
8.1	注意事项	13
8.2	操作流程	13
9	版本历史	14

PUYA CONFIDENTIAL

1 ADC 禁能

1.1 注意事项

- ADC 使能后软件不能禁能，需要复位 ADC 模块，然后重新初始化 ADC，最后启动 ADC。

1.2 操作流程

- 配置 RCC_APBSTR2 寄存器 ADCRST = 1，再配置 ADCRST = 0 复位 ADC 模块；
- 校准 ADC 模块；
- 初始化 ADC 模块；
- 启动 ADC 转换。

1.3 代码示例

```
_HAL_RCC_ADC_FORCE_RESET();
_HAL_RCC_ADC_RELEASE_RESET();/*ADC 复位*/
AdcHandle.Instance = ADC1;
if (HAL_ADCEX_Calibration_Start(&AdcHandle) != HAL_OK) /* ADC 校准 */
{
    Error_Handler();
}
if (HAL_ADC_Init(&AdcHandle) != HAL_OK) /*初始化 ADC*/
{
    Error_Handler();
}
ADC_Cha_SW(); /*ADC 通道初始化*/
HAL_ADC_Start(&AdcHandle);/*启动 ADC 转换*/
```

2 ADC 复位

2.1 注意事项

- 系统上电后硬件不会复位 ADC 模块寄存器, 需要软件复位, 然后重新初始化, 最后启动 ADC。

2.2 操作流程

- 配置 RCC_APBSTR2 寄存器 ADCRST = 1, 再配置 ADCRST = 0 复位 ADC 模块;
- 使能 ADC 时钟
- 校准 ADC 模块;
- 初始化 ADC 模块;
- 启动 ADC 转换。

2.3 代码示例

```
_HAL_RCC_ADC_FORCE_RESET();
_HAL_RCC_ADC_RELEASE_RESET();/*ADC 复位*/
_HAL_RCC_ADC_CLK_ENABLE();/* 使能 ADC 时钟 */
AdcHandle.Instance = ADC1;
if (HAL_ADCEx_Calibration_Start(&AdcHandle) != HAL_OK) /* ADC 校准 */
{
    Error_Handler();
}
if (HAL_ADC_Init(&AdcHandle) != HAL_OK) /*初始化 ADC*/
{
    Error_Handler();
}
ADC_Cha_SW();/*ADC 通道初始化*/
HAL_ADC_Start(&AdcHandle);/*启动 ADC 转换*/
```

3 ADC 校准

3.1 注意事项

- 当 ADC 的工作条件发生改变时 (VCC 改变是 ADC offset 偏移的主要因素, 温度改变次之)
- 第一次使用 ADC 模块前, 必须增加软件校准流程, 如果先配置 ADC 的参数, 再进行校准, 有可能会影响之前的一些参数配置, 所以推荐先校准, 再进行 ADC 的参数配置

3.2 操作流程

- 确认 ADEN = 0;
- 设置 ADCAL = 1;
- 等待到 ADCAL = 0;
- 校准完成后, 参数配置;

3.3 代码示例

```
_HAL_RCC_ADC_FORCE_RESET();
_HAL_RCC_ADC_RELEASE_RESET();
_HAL_RCC_ADC_CLK_ENABLE();
AdcHandle.Instance = ADC1;
if (HAL_ADCEx_Calibration_Start(&AdcHandle) != HAL_OK)
{
    Error_Handler();
}
hadc.Init.ClockPrescaler= ADC_CLOCK_SYNC_PCLK_DIV2;
hadc.Init.Resolution= ADC_RESOLUTION_12B;
hadc.Init.DataAlign= ADC_DATAALIGN_RIGHT;
hadc.Init.ScanConvMode = ADC_SCAN_DIRECTION_FORWARD;
hadc.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
hadc.Init.LowPowerAutoWait = ENABLE;
hadc.Init.ContinuousConvMode = DISABLE;
hadc.Init.DiscontinuousConvMode = DISABLE;
hadc.Init.ExternalTrigConv = ADC_SOFTWARE_START;
hadc.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
hadc.Init.Overrun = ADC_OVR_DATA_OVERWRITTEN;
hadc.Init.SamplingTimeCommon = ADC_SAMPLETIME_41CYCLES_5;
if (HAL_ADC_Init(&AdcHandle) != HAL_OK) /*初始化 ADC*/
{
    Error_Handler();
}
```

4 ADC 仅使用通道 0 的情况

4.1 注意事项

- ADC 在连续模式或不连续模式下，仅使用通道 0 时，必须选择扫描序列向下。

4.2 操作流程

- 当 CFGR1 寄存器中 CONT = 1 或 DISCEN = 1 时（禁止设置 CONT = 1 和 DISCEN = 1），仅使用通道 0 时，配置 SCANDIR = 1。

4.3 代码示例

ADC 连续模式 (CONT = 1, DISCEN = 0)，仅使用通道 0：

```
hadc.Instance = ADC1;
hadc.Init.ClockPrescaler = ADC_CLOCK_SYNC_PCLK_DIV1;
hadc.Init.Resolution = ADC_RESOLUTION_12B;
hadc.Init.DataAlign = ADC_DATAALIGN_RIGHT;
hadc.Init.ScanConvMode = ADC_SCAN_DIRECTION_BACKWARD;
hadc.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
hadc.Init.LowPowerAutoWait = ENABLE;
hadc.Init.ContinuousConvMode = ENABLE;
hadc.Init.DiscontinuousConvMode = DISABLE;
hadc.Init.ExternalTrigConv = ADC_SOFTWARE_START;
hadc.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
hadc.Init.DMAContinuousRequests = DISABLE;
hadc.Init.Overrun = ADC_OVR_DATA_OVERWRITTEN;
hadc.Init.SamplingTimeCommon=ADC_SAMPLETIME_13CYCLES_5;

if (HAL_ADC_Init(&AdcHandle) != HAL_OK) /*初始化 ADC*/
{
    Error_Handler();
}
```

ADC 不连续模式 (CONT = 0, DISCEN = 1)，仅使用通道 0：

```
hadc.Instance = ADC1;
hadc.Init.ClockPrescaler = ADC_CLOCK_SYNC_PCLK_DIV1;
hadc.Init.Resolution = ADC_RESOLUTION_12B;
hadc.Init.DataAlign = ADC_DATAALIGN_RIGHT;
hadc.Init.ScanConvMode = ADC_SCAN_DIRECTION_BACKWARD;
hadc.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
hadc.Init.LowPowerAutoWait = ENABLE;
hadc.Init.ContinuousConvMode = DISABLE;
hadc.Init.DiscontinuousConvMode = ENABLE;
```

```
hadc.Init.ExternalTrigConv = ADC_SOFTWARE_START;
hadc.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
hadc.Init.DMAContinuousRequests = DISABLE;
hadc.Init.Overrun = ADC_OVR_DATA_OVERWRITTEN;
hadc.Init.SamplingTimeCommon=ADC_SAMPLETIME_13CYCLES_5;

if (HAL_ADC_Init(&AdcHandle) != HAL_OK)                //初始化 ADC
{
    Error_Handler();
}
```

5 ADC 连续转换模式

5.1 注意事项

- ADC 在连续转换模式,低功耗等待使能,且通过 DMA 传输时,ADC 时钟源的速度需大于 PCLK 的 1/16,如 PCLK 为 24M 时,则 ADC 的时钟源速度需大于 1.5M.

5.2 操作流程

- 当 CFGR1 寄存器中 CONT = 1, WAIT=1, DMACFG=1, DMAEN=1,

PCLK 为 HSI 且不分频和倍频时,则推荐使用以下时钟

ADC_CLOCK_SYNC_PCLK_DIV1,ADC_CLOCK_SYNC_PCLK_DIV2,ADC_CLOCK_SYNC_PCLK_DIV4,
ADC_CLOCK_SYNC_PCLK_DIV8, ADC_CLOCK_ASYNC_HSI_DIV1, ADC_CLOCK_ASYNC_HSI_DIV2,
ADC_CLOCK_ASYNC_HSI_DIV4, ADC_CLOCK_ASYNC_HSI_DIV8

5.3 代码示例

```
hadc.Instance = ADC1;
hadc.Init.ClockPrescaler = ADC_CLOCK_SYNC_PCLK_DIV1;
hadc.Init.Resolution = ADC_RESOLUTION_12B;
hadc.Init.DataAlign = ADC_DATAALIGN_RIGHT;
hadc.Init.ScanConvMode = ADC_SCAN_DIRECTION_BACKWARD;
hadc.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
hadc.Init.LowPowerAutoWait = ENABLE;
hadc.Init.ContinuousConvMode = ENABLE;
hadc.Init.DiscontinuousConvMode = DISABLE;
hadc.Init.ExternalTrigConv = ADC_SOFTWARE_START;
hadc.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
hadc.Init.DMAContinuousRequests = ENABLE;
hadc.Init.Overrun = ADC_OVR_DATA_OVERWRITTEN;
hadc.Init.SamplingTimeCommon=ADC_SAMPLETIME_13CYCLES_5;
if (HAL_ADC_Init(&AdcHandle) != HAL_OK)                /*初始化 ADC*/
{
    Error_Handler();
}
```

6 ADC 单次模式

6.1 注意事项

- ADC 在单次模式下，转换结束后，需重新使能 ADC 模块 (ADC_EN = 1)，才能开始下一次转换 (ADC_EN 置 1 到 ADSTART 置 1，时间间隔应大于 8 个 ADC 时钟)。

6.2 操作流程

- 初始化 ADC 模块为单次模式；
- 每次启动转换前配置 ADC_EN = 1；
- 配置 ADSTART = 1；
- 开始转换。

6.3 代码示例

Step1: ADC 初始化

```
void ADC_Init()
{
    AdcHandle.Instance = ADC1;
    AdcHandle.Init.ClockPrescaler = ADC_CLOCK_SYNC_PCLK_DIV2;
    AdcHandle.Init.Resolution = ADC_RESOLUTION_12B;
    AdcHandle.Init.DataAlign = ADC_DATAALIGN_RIGHT;
    AdcHandle.Init.ScanConvMode = ADC_SCAN_DIRECTION_BACKWARD;
    AdcHandle.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
    AdcHandle.Init.LowPowerAutoWait = ENABLE;
    AdcHandle.Init.ContinuousConvMode = DISABLE;
    AdcHandle.Init.DiscontinuousConvMode = DISABLE;
    AdcHandle.Init.ExternalTrigConv = ADC_SOFTWARE_START;
    AdcHandle.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;
    AdcHandle.Init.DMAContinuousRequests = DISABLE;
    AdcHandle.Init.Overrun = ADC_OVR_DATA_OVERWRITTEN;
    AdcHandle.Init.SamplingTimeCommon = ADC_SAMPLETIME_3CYCLES_5;

    if (HAL_ADC_Init(&AdcHandle) != HAL_OK) while(1);
}
```

Step2: 软件启动

```
void software_trgmode()
{
    AdcHandle.Instance->CR |= ADC_CR_ADEN;
    HAL_Delay(1);
    AdcHandle.Instance->CR |= ADC_CR_ADSTART;
}
```

Step3: ADC 单次模式转换完整流程

```
while(cnt--)/获取 cnt 个数据*/
{
    software_trgmode();
    if (HAL_ADC_PollForConversion(&AdcHandle,1000) != HAL_OK) while(1);
    *AdcBuff = HAL_ADC_GetValue(&AdcHandle);/*数据保存到 AdcBuff 指定的地址中*/
    AdcBuff++;
}
```

7 ADC 如何利用 VREFINT (1.2V) 测量目标电压值

7.1 注意事项

- ADC 不能直接使用 VREFINT 测量目标电压值，可使用 VREFINT 测量 VCC 值，从而计算出目标电压值。

7.2 操作流程

- 以测量通道 0 的电压值为例；
- 先读出内部参考电压的 ADC 测量结果，记为 ADvrefint，再读出通道 0 的 ADC 测量结果，记为 ADch0；
- 要测量的电压为 $V_{ch0} = VREFINT * (ADch0 / ADvrefint)$ ，VREFINT 为 1.2V。

7.3 代码示例

```
float T_VCC;
uint16_t aADCxConvertedData[32];
AdcBuff = aADCxConvertedData;

if (HAL_ADC_Start(&AdcHandle) != HAL_OK) while(1); //启动 ADC
i = 2;
while(i--)
{
    if (HAL_ADC_PollForConversion(&AdcHandle, 1000) != HAL_OK) while(1);
    *AdcBuff = HAL_ADC_GetValue(&AdcHandle);
    AdcBuff++;
}
T_VCC = (aADCxConvertedData[0]*1.2)/aADCxConvertedData[1];
```

8 AD 口配置注意事项

8.1 注意事项

- 所有 AD 口输入电压不能超过 VCC，即使不配置 AD 功能也不能超过 VCC。

8.2 操作流程

- 将超 VCC 的 AD 口改为非 AD 口；

9 版本历史

版本	日期	更新记录
V0.1	2021.10.15	初版
V1.0	2022.06.22	增加 6. ADC 如何利用 VREFINT (1.2V) 测量目标电压值
V1.1	2022.10.21	添加校准的顺序和连续转换的注意事项
V1.2	2023.02.28	增加 AD 口配置注意事项



Puya Semiconductor Co., Ltd.

IMPORTANT NOTICE

Puya Semiconductor reserves the right to make changes without further notice to any products or specifications herein. Puya Semiconductor does not assume any responsibility for use of any its products for any particular purpose, nor does Puya Semiconductor assume any liability arising out of the application or use of any its products or circuits. Puya Semiconductor does not convey any license under its patent rights or other rights nor the rights of others.