HOW TO MINIMIZE THE ST7 POWER CONSUMPTION

By MCD Application Team

1 INTRODUCTION

The purpose of this document is to explain the different low power modes available on ST7 devices and the ways to minimise power consumption. Many applications will have strict power requirements, and there are several methods of lowering the rate of power consumption without sacrificing performance. Calculating the predicted power use is important to characterize the system's power supply requirements. The ST7 can be put into one of several low power modes by setting some bits in some registers. The utility of these low power modes depends on the specific application.

The basic explanation of this note is based on ST72F324, but is applicable to all ST7 general purpose devices. Please refer section "Examples" to see more information on additional devices (ST7FLITE0).

2 POWER CONSUMPTION FACTORS

CMOS digital logic device power consumption is affected by supply voltage and clock frequency. These parameters can be adjusted to realize power savings, and are readily controlled by the designer. In CMOS digital logic devices, power consumption is directly proportional to clock frequency and power supply squared.

$$power = CV^2f$$

where: C is CMOS load capacitance, V is supply voltage, and f is clock frequency.

The amount of current used in CMOS logic is directly proportional to the voltage of the power supply. Thus, power consumption may be reduced by lowering the supply voltage to the device. Power consumption depends on the number of active peripherals. The greater the number of active peripherals, the more power will be consumed. Power consumption also depends on, whether the oscillator is On or Off and whether the CPU is On or Off. It also depends on PLL On/Off, CSS enabled/disabled and LVD On/Off.

Power Consumption is based on which mode a particular application is running. For example, in ST7, "HALT" mode is the lowest power consumption mode without availability of Real Time Clock and "ACTIVE-HALT" mode is the lowest power consumption mode with Real Time Clock available. To reduce the power consumption, clock frequency can be reduced whenever fast processing is not required by the application.

1 INTRODUCTION 1

2 POWER CONSUMPTION FACTORS 1

3 POWER MANAGEMENT MODES 4

3.1 STANDARD RUN MODE 4

3.2 SLOW MODE 4

3.3 WAIT MODE 4

3.4 SLOW WAIT MODE 5

3.5 ACTIVE-HALT MODE 5

3.6 HALT MODE 5

3.7 SUMMARY 6

4 EXAMPLES 6

5 POWER MANAGEMENT TIPS 6

6 APPENDIX A: ST72F324 STANDARD EXAMPLES 7

6.1 EXAMPLE 1: STATIC MEASUREMENT 7

6.1.1 Measurement Configuration 7
6.1.2 Consumption 7
6.1.2.1 Run Mode 7
6.1.2.2 Slow Mode 8
6.1.2.3 Wait Mode 8
6.1.2.4 Slow-Wait Mode 9
6.1.2.5 Active-Halt Mode 10
6.1.2.6 Halt Mode 10
6.1.3 Conclusion 10

6.2 EXAMPLE 2: APPLICATION WITH PERIODIC WAKE-UP 11

6.2.1 Measurement Configuration 11
6.2.2 Consumption 11
6.2.2.1 Wait Mode 12
6.2.2.2 Slow-Wait Mode 12
6.2.2.3 Halt Mode 13
6.2.3 Conclusion 13

6.3 EXAMPLE 3: APPLICATION WITH PERIODIC WAKEUP AND WATCHDOG 13

6.3.1 Measurement Configuration 14

6.3.2 Application 15
6.3.3 Conclusion 15

7 APPENDIX B: ST7FLITE0 STANDARD EXAMPLES 16

7.1 EXAMPLE 1: STATIC MEASUREMENT 16

7.1.1 Measurement Configuration 16
7.1.2 Consumption 16
7.1.2.1 Run Mode 17
7.1.2.2 Slow Mode 17
7.1.2.3 Wait Mode 18
7.1.2.4 Slow-Wait Mode 18
7.1.2.5 Active-Halt Mode 19
7.1.2.6 Halt Mode 19
7.1.3 Conclusion 19

7.2 EXAMPLE 2: APPLICATION WITH PERIODIC WAKE-UP 19

7.2.1 Measurement Configuration 20
7.2.2 Consumption 20
7.2.2.1 Wait Mode 20
7.2.2.2 Slow-Wait Mode 21
7.2.2.3 Halt Mode 21
7.2.3 Conclusion 22

7.3 EXAMPLE 3: APPLICATION WITH PERIODIC WAKEUP AND WATCHDOG 22

7.3.1 Measurement Configuration 22
7.3.2 Application 23
7.3.3 Conclusion 24

3 POWER MANAGEMENT MODES

The ST7 can run in the following six main modes:

3.1 STANDARD RUN MODE

This mode is the normal operation of any MCU, where

$f_{CPU} = f_{OSC2} = f_{OSC}/2$ (when PLL is disabled by OPTION BYTE).

$f_{OSC2} = f_{OSC}*2$ (when PLL is enabled by OPTION BYTE).

So, the consumption varies depending on whether the PLL is disabled or enabled.

3.2 SLOW MODE

This mode is controlled by three bits in the MCCSR register: the SMS bit which enables or disables Slow mode and two CPx bits which select the internal slow frequency (f_{CPU}).

In this mode, the master clock frequency (f_{OSC2}) can be divided by 2, 4, 8 or 16.

The CPU and peripherals are clocked at this lower frequency (f_{CPU}). The configuration for clock frequency is:

$f_{CPU} = f_{OSC2}$ / 2, 4, 8, 16

$= f_{OSC}$ / 4, 8, 16 or 32 (when PLL is disabled by OPTION BYTE).

Uses:

- To reduce the power consumption by decreasing the internal clock in the device.

- To adapt the internal clock frequency to the available supply voltage.

3.3 WAIT MODE

In this mode, the CPU is stopped and the peripherals are still running at standard f_{CPU}. It is selected by calling the ‘WFI’ instruction. All peripherals remain active.

During WAIT mode, the I[1:0] bits in the CC register are forced to ‘10’, to enable all interrupts. All other registers and memory remain unchanged. The MCU remains in WAIT mode until an interrupt or RESET occurs, whereupon the Program Counter branches to the starting address of the interrupt or Reset service routine. The MCU will remain in WAIT mode until a Reset or an Interrupt occurs, causing it to wake up.

Uses:

- To place the MCU in low power consumption mode by stopping the CPU.

- External interrupt capability with all peripherals remaining active

3.4 SLOW WAIT MODE

In this mode, the CPU is stopped and the peripherals are still running at the f_{CPU} defined for SLOW mode. It is activated when entering WAIT mode while the device is already in SLOW mode.

Uses:

- To place the MCU simultaneously in Slow mode and Wait mode to reduce the power consumption.

3.5 ACTIVE-HALT MODE

In this mode, the CPU and the peripherals are stopped, but the oscillator is still running. Peripherals clocked with an external clock source can still be active.

It is selected by calling the "HALT" instruction when the MCCSR - OIE bit is set. "HALT" forces the I[1:0] bits in the CC register to '10' to enable interrupts. The CPU clock is stopped till a Reset or MCC/RTC or CSS or other specific interrupt occurs.

To wake-up from ACTIVE-HALT, a Reset or MCC/RTC or CSS or other specific interrupt must occur. Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits in the CC register are set to the current software priority level of the interrupt routine and recovered when the CC register is popped. The safeguard against staying locked in ACTIVE-HALT mode is provided by the oscillator interrupt.

As soon as the interrupt capability of one of the oscillators is selected (MCCSR.OIE bit set), entering ACTIVE-HALT mode while the Watchdog is active does not generate a RESET. Because the watchdog remains active, this means that the MCU cannot spend more than a defined delay in ACTIVE-HALT mode.

Uses:

- To place the MCU in the lowest power consumption mode with Real Time Clock available.
- The CPU and Peripherals (Peripheral clocked with external clock source can still be active) are OFF.
- To keep a wake-up time base, the Real Time Clock Main Clock Controller is running.

3.6 HALT MODE

In this mode, the oscillator is turned off. Peripherals clocked with an external clock source can still be active. Halt mode is selected by calling a "HALT" Instruction while the MCCSR - OIE bit is cleared. "HALT" forces the I[1:0] bits in the CC register to '10' to enable interrupts.

The CPU clock is stopped till a Reset or a specific interrupt (with "exit from Halt" capability) occurs. To wake-up the MCU from Halt mode (when the Watchdog is active or when the

Watchdog is inactive and the WDGHALT option bit is disabled.), a Reset or specific interrupt must occur. Before servicing an interrupt, the CC register is pushed on the stack. The I[1:0] bits in the CC register are set to the current software priority level of the interrupt routine and recovered when the CC register is popped. When the Watchdog is active and the WDGHALT option bit is enabled, a Watchdog reset is generated.

Uses:

– To place the MCU in the lowest power consumption mode without Real Time Clock.

– The CPU and Peripherals (Peripheral clocked with external clock source can still be active) are OFF.

3.7 SUMMARY

Table 1. Summary Table

ST7 Modes	Oscillator/CPU/Peripheral Status					
	Oscillator	**CPU**	**Peripherals**	**RTC**	**MCCSR-OIE Bit**	**MCCSR-SMS Bit**
Run	On	On	On	Available	X	Reset
Slow	On	On	On	Available	X	Set
Wait	On	Off	On	Available	X	Reset
Slow-Wait	On	Off	On	Available	X	Set
Active-Halt	On	Off	Off	Available	Set	X
Halt	Off	Off	Off	Not Available	Reset	X

4 EXAMPLES

This section provides standard methods to achieve minimum power consumption in a particular ST7 application for following different microcontrollers, which can be used as a reference during application development. For user reference it also provides data values measured in lab for described applications presented as examples for different devices. CLICK on required device to see more information on

- Appendix A: ST72F324 Standard Examples

- Appendix B: ST7FLITE0 Standard Examples

Note: The values provided in this application note are typical only, measured on a small number of devices.

5 POWER MANAGEMENT TIPS

– If you are not using the ADC, SPI, SCI or timers in the application, switch them off.

– All the port pins should be push pull output at low level.

– All I/O ports should be connected to an external pull-up or pull-down to avoid leakage due to floating inputs.

– Use Wait mode if you need external interrupt capability in low power mode and if peripherals are to be remain active.

– Use the appropriate V_{DD} value. The V_{DD} value must not be greater than the required value because higher the V_{DD} value, the more power will be consumed.

– Configure the OSCRANGE[2:0] option bits for the minimum frequency range. For example, if you use 8 MHz oscillator frequency, you must select the OSCRANGE[2:0] option bits for medium speed resonator 4/8 MHz, not for high speed resonator 8/16 MHz.

6 APPENDIX A: ST72F324 STANDARD EXAMPLES

This section provides examples of how to minimise the consumption in a particular ST7 application for device ST72F324.(Use ST72F324 datasheet for a reference)

6.1 EXAMPLE 1: STATIC MEASUREMENT

This example provides a static method of measuring the current consumed by the microcontroller in different modes and without any I/O activity. The measurement is done using the following configuration.

6.1.1 Measurement Configuration

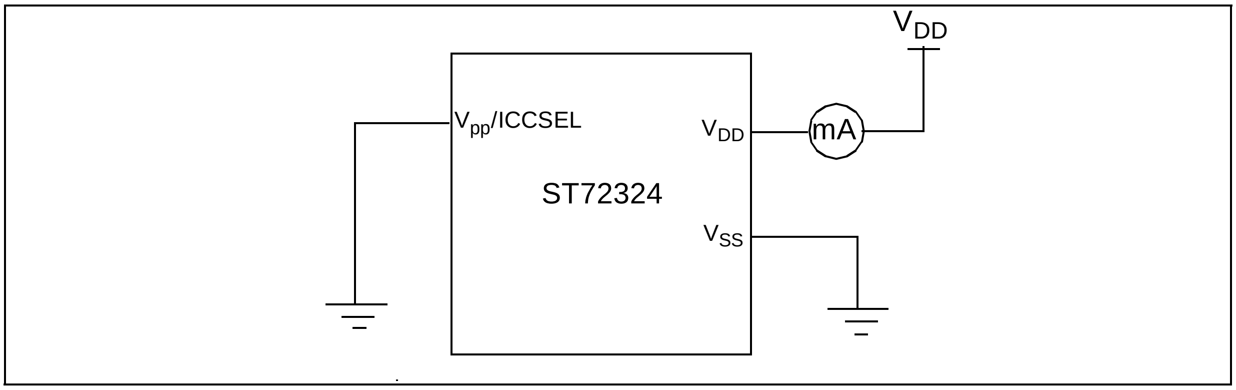
– All ports have been set as Push-Pull Outputs at low level.

– All other peripherals are in reset configuration.

– After this, MCU is put into different modes by calling different instructions and by setting some bits (in the MCCSR register).

– In Option Byte, PLL*2 is disabled, CSS is also disabled and LVD is Off.

Figure 1. Hardware Setup



6.1.2 Consumption

The consumption mainly depends on the mode selected and the CPU frequency.

6.1.2.1 Run Mode

Methodology:

A resonator oscillator is used with f_{osc} at 4 / 8 MHz. PLL*2 is disabled (So, f_{osc2} is $f_{osc}/2$). All peripherals are in reset configuration except the ports. So, the SMS bit in the MCCSR is reset to 0. The CP1 and CP0 bits in the MCCSR are also reset to 0 and hence f_{CPU} is f_{osc2}. All ports are set as Push-Pull Outputs with low level.

Measurements:

Table 2. Consumption I_{DD} (RUN Mode) at T_A = + 25 °C

f_{OSC}(MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
4	**2**	3.90 mA	4.38 mA	4.91 mA
8	**4**	5.88 mA	6.67 mA	7.42 mA

6.1.2.2 Slow Mode

Methodology: A resonator oscillator is used with f_{osc} at 4 / 8 MHz. PLL*2 is disabled. So, f_{osc2} is $f_{osc}/2$. All peripherals are in reset configuration except the ports and the SMS bit in the MCCSR is set to 1. The CP1 and CP0 bits in the MCCSR are set to an appropriate value. Hence f_{CPU} is f_{osc} /4, $f_{osc}/8$, $f_{osc}/16$, $f_{osc}/32$. All ports are set as Push-Pull Outputs with low level.

Measurements:

Table 3. Consumption I_{DD} (SLOW Mode) at T_A = + 25 °C

f_{OSC}(MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
4	**0.125**	1.33 mA	1.51 mA	1.70 mA
4 / 8	**0.25**	1.67 mA	1.93 mA	2.18 mA
4 / 8	**0.5**	2.25 mA	2.54 mA	2.88 mA
4 / 8	**1**	2.87 mA	3.25 mA	3.64 mA
8	**2**	3.88 mA	4.46 mA	4.95 mA

6.1.2.3 Wait Mode

Methodology: A resonator oscillator is used with f_{osc} at 4 / 8 MHz. PLL*2 is disabled. So, f_{osc2} is $f_{osc}/2$. All peripherals are in reset configuration except the ports. So, the SMS bit in the MCCSR is reset to 0. The CP1 and CP0 bits in the MCCSR are also reset to 0 and hence f_{CPU} is f_{osc2}. All ports are set as Push-Pull Outputs with low level.

Measurements:

Table 4. Consumption I_{DD} (Wait Mode) at T_A = + 25 °C

f_{OSC}(MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
4	**2**	3.49 mA	3.91 mA	4.39 mA
8	**4**	5.20 mA	5.92 mA	6.62 mA

6.1.2.4 Slow-Wait Mode

Methodology: A resonator oscillator is used with f_{osc} at 4 / 8 MHz. PLL*2 is disabled. So, f_{osc2} is f_{osc}/2. All peripherals are in reset configuration except the ports and the SMS bit in the MCCSR is set to 1. The CP1 and CP0 bits in the MCCSR are set to an appropriate value. Hence f_{CPU} is f_{osc} /4, f_{osc}/8, f_{osc}/16, f_{osc}/32. All ports are set as Push-Pull Outputs with low level.

Measurements:

Table 5. Consumption I_{DD} (Slow-Wait Mode) at T_A = + 25 °C

f_{OSC}(MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
4	**0.125**	1.28 mA	1.44 mA	1.63 mA
4 / 8	**0.25**	1.59 mA	1.85 mA	2.10 mA
4 / 8	**0.5**	2.12 mA	2.45 mA	2.77 mA
4 / 8	**1**	2.69 mA	3.02 mA	3.39 mA
8	**2**	3.56 mA	3.99 mA	4.39 mA

6.1.2.5 Active-Halt Mode

Methodology: A resonator oscillator is used with f_{osc} at 4 / 8 MHz. PLL*2 is disabled. So, f_{osc2} is $f_{osc}/2$. All peripherals are in reset configuration except the ports. All ports are set as Push-Pull Outputs with low level. The MCCSR-OIE bit is set to 1 and the TB1 & TB0 of MCCSR bits are set to 1. The MCC/RTC interrupt is used as the wake-up interrupt. The oscillator is ON, so the total consumption also includes the oscillator consumption.

Measurements:

Table 6. Consumption I_{DD} (Active-Halt Mode) at T_A = + 25 °C

f_{OSC}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
4	1.06 mA	1.20 mA	1.36 mA
8	1.19 mA	1.35 mA	1.53 mA

6.1.2.6 Halt Mode

Methodology: A resonator oscillator is used with F_{osc} at 4 / 8 MHz. PLL*2 is disabled. So, F_{osc2} is $F_{osc}/2$. All peripherals are in reset configuration except the ports. All ports are set as Push-Pull Outputs with low level.

Measurements:

Table 7. Consumption I_{DD} (Halt Mode) at T_A = + 25 °C

f_{OSC}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
4	0.1 µA	0.1 µA	0.1 µA
8	0.1 µA	0.1 µA	0.1 µA

6.1.3 Conclusion

Current consumption depends on V_{DD}. Consumption increases with the increase of V_{DD} and f_{OSC}. In Active-Halt mode, the time base for the MCC/RTC interrupt does not affect the current consumption in this particular application. The oscillator is ON in Active-Halt mode, so the total consumption also includes the oscillator consumption. In Halt mode, current consumption is independent of V_{DD} and f_{OSC}.

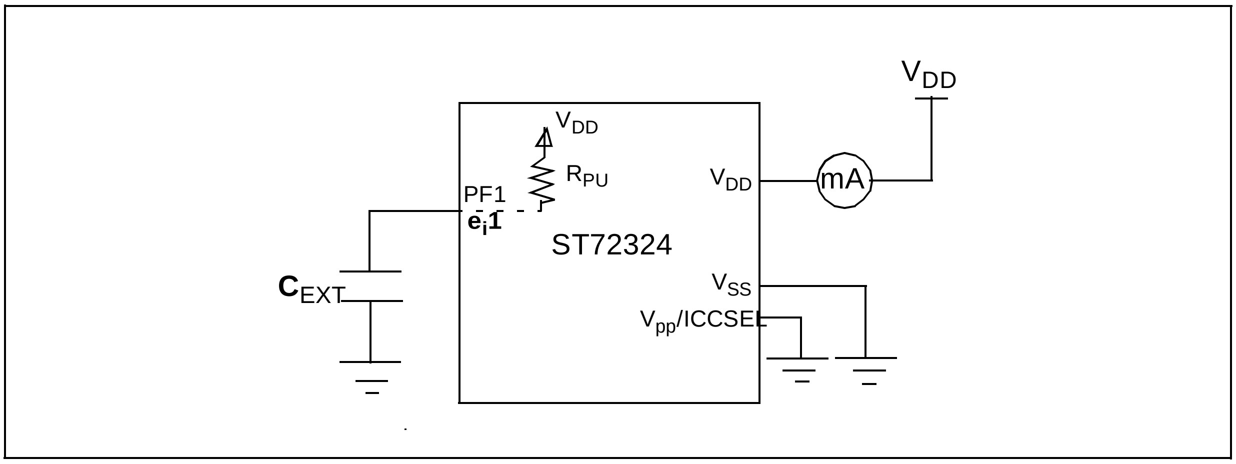
6.2 EXAMPLE 2: APPLICATION WITH PERIODIC WAKE-UP

This example provides a method for measuring the average current consumed by the micro in different modes while it cycles continuously through “wake-up” and “sleep” periods. The measurement is done using the following configuration.

6.2.1 Measurement Configuration

- The PF1 port (through which an external interrupt is applied to MCU) is configured as pull up interrupt.
- The sensitivity of the interrupt is configured as rising edge.
- Then the capacitor is charged and immediately the MCU is put into the corresponding mode (Wait, Slow-Wait, Halt) by calling a “WFI” or “HALT” instruction and setting some bits.
- As soon as the capacitor charges to 0.7V_{DD}, the MCU recognizes it as external interrupt and comes out of low power mode.
- The capacitor is then discharged by setting the PF1 port to Push-Pull Output at low level and a small delay is then provided to let the capacitor discharge fully before charging it the next time.
- The MCU is again put into the corresponding low power mode and the same process is repeated. The current consumption is measured in continuous interrupted mode.
- In Option Byte, PLL*2 is disabled, CSS is also disabled and LVD is Off.

Figure 2. Hardware Setup



6.2.2 Consumption

The consumption mainly depends on how frequently the MCU wakes up from different low power modes. The Wake-Up time in turn depends upon the RC time constant. The value of the C_{ext} is fixed but the internal pull up resistance value varies depending upon the power supply of the MCU. R_{PU} decreases with the increase of V_{DD}.

The variation of R_{PU} with temperature and V_{DD} is given in the datasheet.

6.2.2.1 Wait Mode

Methodology: A resonator oscillator is used with f_{osc} at 4 / 8 MHz. PLL*2 is disabled. So, f_{osc2} is $f_{osc}/2$. All peripherals are in reset configuration except the ports. So, the SMS bit in the MCCSR is reset to 0. The CP1 and CP0 bits in the MCCSR are also reset to 0 and hence F_{CPU} is f_{osc2}. IS21 bit of EICR is cleared and IS20 bit of EICR is set to 1. All Ports are in Push-Pull Output with low level except PF1 (which is used for charging and discharging the capacitor).

Measurements:

Table 8. Consumption I_{DD} (Wait Mode) for C_{ext} = 1 uF at T_A = + 25 °C

f_{OSC}(MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
4	**2**	3.64 mA	4.06 mA	4.63 mA
8	**4**	5.44 mA	6.16 mA	6.91 mA

6.2.2.2 Slow-Wait Mode

Methodology: A resonator oscillator is used with f_{osc} at 4 / 8 MHz. PLL*2 is disabled. So, f_{osc2} is $f_{osc}/2$. All peripherals are in reset configuration except the ports and the SMS bit in the MCCSR is set to 1. The CP1 and CP0 bits in the MCCSR are set accordingly. Hence f_{CPU} is f_{osc} /4, f_{osc}/8, f_{osc}/16, f_{osc}/32. The IS21 bit in the EICR is cleared and IS20 bit in the EICR is set to 1. All ports are set as Push-Pull Outputs with low level except PF1 (which is used for charging and discharging the capacitor).

Measurements:

Table 9. Consumption I_{DD} (Slow-Wait Mode) for C_{ext} = 1 uF at T_A = + 25 °C

f_{OSC}(MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
4	**0.125**	1.38 mA	1.55 mA	1.78 mA
4 / 8	**0.25**	1.68 mA	1.93 mA	2.19 mA
4 / 8	**0.5**	2.24 mA	2.51 mA	2.88 mA
4 / 8	**1**	2.80 mA	3.17 mA	3.53 mA
8	**2**	3.68 mA	4.16 mA	4.68 mA

6.2.2.3 Halt Mode

Methodology: A resonator oscillator is used with f_{osc} at 8 MHz. PLL*2 is disabled. So, f_{osc2} is $f_{osc}/2$. All peripherals are in reset configuration except the ports. The IS21 bit in the EICR is cleared and the IS20 bit in the EICR is set to 1. All ports are set as Push-Pull Outputs with low level except PF1 (which is used for charging and discharging the capacitor).

Measurements:

Table 10. Consumption I_{DD} (Halt Mode) for C_{ext} = 1 uF at T_A = + 25 °C

f_{OSC}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
8	360 µA	580 µA	840 µA

6.2.3 Conclusion

If V_{DD} increases, R_{PU} decreases. So, the RC time constant also decreases for a fixed value of C_{ext}. So, the wake-up time decreases. Hence the wake-up frequency increases with the increase of V_{DD}.

A delay after wake-up by an interrupt exists only in Halt mode, because in all other low power modes (except Halt mode) the oscillator is ON. So, to wake up from HALT mode, the MCU needs 256/4096 CPU cycles delay for the oscillator startup, during this time it consumes some current.

* Wake-Up frequency is the external interrupt frequency (Wake-Up time is the time after which the MCU is woken up from the different modes (Wait, Slow-Wait, Active-Halt, Halt)).

6.3 EXAMPLE 3: APPLICATION WITH PERIODIC WAKEUP AND WATCHDOG

For this example, we assume that for the application the Watchdog Timer (WDG) has to remain active and that the microcontroller has to stay in low power mode for an undefined time period. Therefore, Halt mode can not be used, as the “Halt” instruction resets the microcontroller when the WDG is active and WDGHALT option bit in the option byte is set to 1. When the WDGHALT option bit in the option byte is 0 and WDG is active, a “Halt” instruction doesn’t generate a Watchdog reset. In this case, the MCU enters Halt mode. The Watchdog counter is decremented once and then stops counting and is no longer able to generate a watchdog reset until the MCU receives an external interrupt or reset. Active-Halt mode can also not be used, because, the device cannot spend more than a defined period in this power saving mode. Consequently, Wait mode is used, associated with Slow mode configured for the slowest frequency. This slow mode clock (f_{CPU}) drives the active peripherals and the CPU when it is woken-up. Of course, to reduce power consumption further, Halt mode should be used wherever possible.

In terms of applications, users usually try to optimize the power consumption during the wait loop in the main program. In this configuration the microcontroller is mostly idle, waiting for an external event, or regularly woken-up by an internal timer event. In our case, we decided to put the ST7 core in Wait For Interrupt state, with one timer active to regularly wake up the core. This enables the core to refresh the WDG to prevent a watchdog reset. The clock driving the active peripherals and the core when it is active corresponds to the slowest clock available. It is the oscillator clock divided by 2 then divided by 16. So the clock driving the peripherals and core (f_{CPU}) is equal to the crystal value (f_{OSC}) divided by 32. The measurement is done using the following configuration.

6.3.1 Measurement Configuration

In this mode, the MCU is configured as below, but of course in the application this configuration may differ, in particular the I/O port used.

- The optimum Slow mode is selected with the divided external clock factor equal to 32. This clock (f_{CPU}) drives the active peripherals (WDG and Timer A) and the core when it is active. In the Miscellaneous Register (MISCR) the SMS and CP1 & CP0 bits must be set.

- All I/O ports are connected to an external pull-up or pull-down to avoid leakage due to floating inputs.

- The Watchdog Timer (WDG) is active with a maximum time-out period. The WDG Control Register (WDGCR) must be loaded with the value FFh and the TB1 & TB0 bits must be set.

- One 16-bit timer is used to wake-up the core at regular intervals. The timer event period must be adjusted to its maximum value to allow the core to refresh the WDG and to avoid a WDG reset. The timer clock equals $f_{CPU}/8$, in the Control Register (CR2), CC1 bit must be set & CC0 bits must be cleared.

- The external clock is selected on the unused 16-bit timer with a continuous level on the external clock pin. In the Control Register (CR2) CC1, CC0 bits must be set.

Note: If timer A is used to regularly wake-up the WDG, the user only has to select the external clock on timer B. The continuous level on the timer input clock is already done internally, as there is no external I/O connected to the internal Timer B circuit. It stops the counter on timer B, because if external clock is not available, programming the external clock configuration stops the counter.

On the contrary, if Timer B is used to wake-up the micro, the user has to select the external clock on Timer A and to fix a continuous level on the external clock pin. But be aware, on Timer A there is only one input capture and one output compare instead of two on Timer B.

– The main loop contains a Wait For Interrupt “WFI” instruction.

– Continuous low level on each PWM output (reset state)

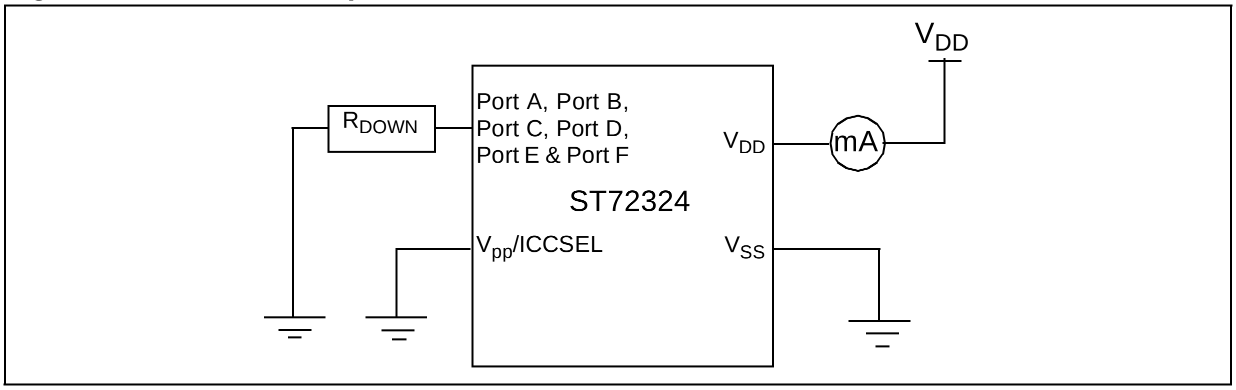
– The A/D converter must be switched off. In the A/D Control/Status Register (CSR), the ADON bit must be cleared.

– The SPI and SCI are unused.

– In Option Byte, PLL*2 is disabled, CSS is also disabled and LVD is Off.

For the ST7 software configuration, please refer to your datasheet.

Figure 3. Hardware Setup



6.3.2 Application

Methodology: A resonator oscillator is used with f_{osc} at 4 / 8 MHz. PLL*2 is disabled. So, f_{osc2} is $f_{osc}/2$. All peripherals are in reset configuration and the SMS bit in the MCCSR is set to 1. The CP1 and CP0 bits in the MCCSR are set to 1. Hence f_{CPU} is $f_{osc}/32$. The TB1 and TB0 bits in the MCCSR are set to 1. All I/O ports are connected to an external pull-down. OCIE bit of TACR1 is set to 1. CC1 bit of TACR2 is set to 1 and CC0 bit of TACR2 is cleared. CC1 and CC0 bits of TBCR2 are set to 1.

Measurements:

The measurement of current (I_{DD}) is done directly on the V_{DD} pin. The microcontroller is in Wait mode and is configured as described above, with the WDG and one 16-bit Timer active. The values given in Table11 correspond to typical measured values. They are not maximum or minimum values.

Table 11. Consumption I_{DD} at T_A = + 25 °C

f_{OSC}(MHz)	f_{CPU}(kHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
4	**125**	1.26 mA	1.43 mA	1.62 mA
8	**250**	1.54 mA	1.80 mA	2.02 mA

6.3.3 Conclusion

Current consumption depends on V_{DD}. Current consumption increases with the increase of V_{DD} and F_{OSC}. In this particular application, Timer B is disabled. So, by disabling the Timer B, current consumption reduces. CPU frequency and Timer frequency also decides the current consumption.

7 APPENDIX B: ST7FLITE0 STANDARD EXAMPLES

This section provides examples of how to minimise the consumption in a particular ST7 application for device ST7FLITE09.(Use ST7LITE0 datasheet for a reference)

7.1 EXAMPLE 1: STATIC MEASUREMENT

This example provides a static method of measuring the current consumed by the microcontroller in different modes and without any I/O activity. The measurement is done using the following configuration.

7.1.1 Measurement Configuration

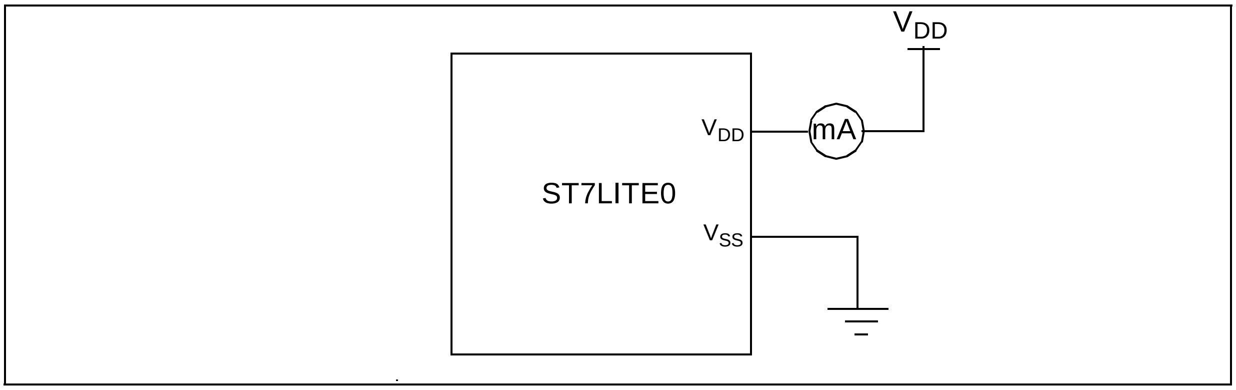
– All ports have been set as Push-Pull Outputs at low level.

– All other peripherals are in reset configuration.

– After this, MCU is put into different modes by calling different instructions and by setting some bits (in the MCCSR & ATCSR registers).

– In Option Byte, PLL is disabled, Internal RC Oscillator is OFF, LVD is OFF & No Reset when Halt is selected.

Figure 4. Hardware Setup



7.1.2 Consumption

The consumption mainly depends on the mode selected, CPU frequency & Supply voltage.

7.1.2.1 Run Mode

Methodology: An external CLK source, up to 16MHz is used on pin CLKIN. PLL is disabled and internal RC Oscillator is OFF. So, f_{OSC} is CLKIN / 2. All peripherals are in reset configuration except the I/O Ports. At reset the SMS bit in the MCCSR is 0, hence f_{CPU} is f_{OSC}. All Ports are in Push-Pull Outputs with low level.

Measurements:

Table 12. Consumption I_{DD} (RUN Mode) at T_A = + 25 °C

External CLK on CLKIN Pin (MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
16	**8**	4.5 mA	5.17 mA	5.86 mA
8	**4**	2.32 mA	2.68 mA	3.07 mA

7.1.2.2 Slow Mode

Methodology: An external CLK source, up to 16MHz is used on pin CLKIN. PLL is disabled and internal RC Oscillator is OFF. So, f_{OSC} is CLKIN / 2. All peripherals are in reset configuration except the I/O Ports and the SMS bit in the MCCSR is set to 1, hence f_{CPU} is f_{OSC}/32. All Ports are in Push-Pull Outputs with low level.

Measurements:

Table 13. Consumption I_{DD} (SLOW Mode) at T_A = + 25 °C

External CLK on CLKIN Pin (MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
16	**0.25**	417 µA	490 µA	571 µA
8	**0.125**	274 µA	320 µA	372 µA
2	**0.03125**	142 µA	161 µA	192 µA

7.1.2.3 Wait Mode

Methodology: An external CLK source, up to 16MHz is used on pin CLKIN. PLL is disabled and internal RC Oscillator is OFF. So, f_{OSC} is CLKIN / 2. All peripherals are in reset configuration except the I/O Ports. At reset the SMS bit in the MCCSR is 0, hence f_{CPU} is f_{OSC}. All Ports are in Push-Pull Outputs with low level.

Measurements:

Table 14. Consumption I_{DD} (Wait Mode) at T_A = + 25 °C

External CLK on CLKIN Pin (MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
16	**8**	1434 µA	1685 µA	1920 µA
8	**4**	907 µA	1054 µA	1205 µA
2	**1**	315 µA	365 µA	433 µA

7.1.2.4 Slow-Wait Mode

Methodology: An external CLK source, up to 16MHz is used on pin CLKIN. PLL is disabled and internal RC Oscillator is OFF. So, f_{OSC} is CLKIN / 2. All peripherals are in reset configuration except Ports and the SMS bit in the MCCSR is set to 1, hence f_{CPU} is f_{OSC}/32. All Ports are in Push-Pull Outputs with low level.

Measurements:

Table 15. Consumption I_{DD} (Slow-Wait Mode) at T_A = + 25 °C

External CLK on CLKIN Pin (MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
16	**0.25**	337 µA	402 µA	475 µA
8	**0.125**	230 µA	270 µA	316 µA
2	**0.03125**	128 µA	146 µA	170 µA

7.1.2.5 Active-Halt Mode

Methodology: An external CLK source, up to 16MHz is used on pin CLKIN. PLL is disabled and internal RC Oscillator is OFF. So, f_{OSC} is CLKIN / 2. All peripherals are in reset configuration except the I/O Ports. All Ports are in Push-Pull Outputs with low level. Set ATCSR- CK1 bit to 0 & CK0 & OVFIE bits to 1. MCU is woken-up by AT TIMER Overflow Interrupt. The oscillator is ON, so the total consumption includes the oscillator consumption also.

Measurements:

Table 16. Consumption I_{DD} (Active-Halt Mode) at T_A = + 25 °C

External CLK on CLKIN Pin (MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
16	**8**	366 µA	428 µA	501 µA
8	**4**	238 µA	277 µA	321 µA
2	**1**	122 µA	136 µA	157 µA

7.1.2.6 Halt Mode

Methodology: An external CLK source, up to 16MHz is used on pin CLKIN. PLL is disabled and internal RC Oscillator is OFF. So, f_{OSC} is CLKIN / 2. All peripherals are in reset configuration except Ports. All Ports are in Push-Pull Outputs with low level.

Measurements:

Table 17. Consumption I_{DD} (Halt Mode) at T_A = + 25 °C

External CLK on CLKIN Pin (MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
16	**8**	0.1 µA	0.1 µA	0.1 µA
8	**4**	0.1 µA	0.1 µA	0.1 µA

7.1.3 Conclusion

Current consumption depends on V_{DD}. Consumption increases with the increase of V_{DD} and External clock frequency. The oscillator is ON in Active-Halt mode, so the total consumption includes the oscillator consumption also. In Halt mode, current consumption is independent of V_{DD} and External clock frequency.

7.2 EXAMPLE 2: APPLICATION WITH PERIODIC WAKE-UP

This example provides a method for measuring the average current consumed by the micro in different modes while it cycles continuously through “wake-up” and “sleep” periods. The measurement is done using the following configuration.

7.2.1 Measurement Configuration

– The port PA0 (through which external interrupt is applied to MCU) is configured as pull up interrupt.

– The sensitivity of the interrupt is configured as rising edge.

– Then the capacitor is charged and immediately the MCU is put into the corresponding mode (Wait, Slow-Wait, Halt) by calling “WFI” or “HALT” instruction and setting some bits.

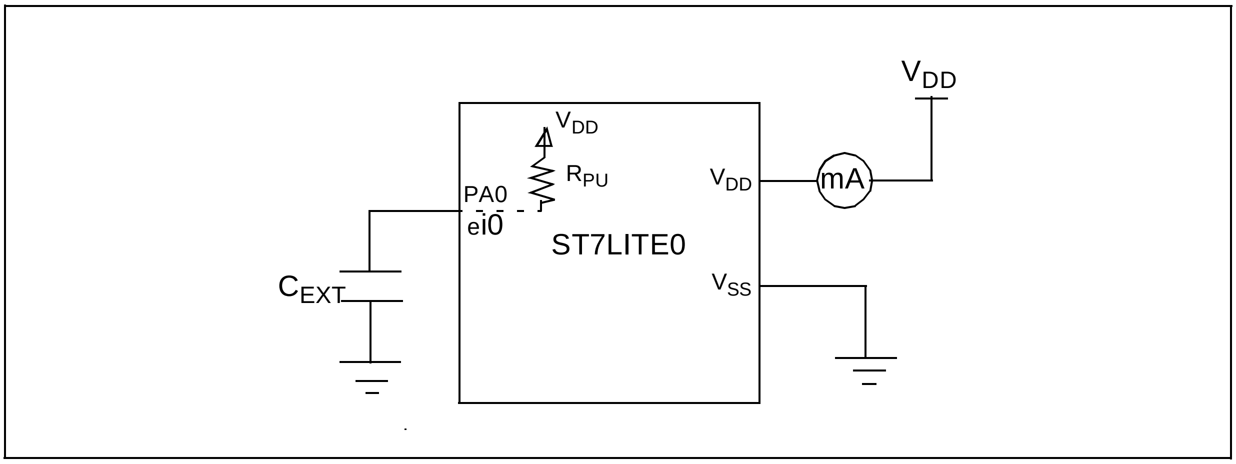
– As soon as the capacitor charges to 0.7Vdd, the MCU takes it as external interrupt and comes out of that mode.

– The capacitor is then discharged by setting PA0 port to Push-Pull Output at low level and small delay is then provided to let the capacitor discharge fully before charging it next time.

– The MCU is again put into corresponding mode and the same process is repeated. The Measurement of current consumption is done in continuous interrupted mode.

– In Option Byte, PLL is disabled, Internal RC Oscillator is OFF, LVD is Off & No Reset when Halt is selected.

Figure 5. Hardware Setup



7.2.2 Consumption

The consumption mainly depends on the Wake-Up frequency of the MCU from different modes. The Wake-Up time in turn depends upon the RC time constant. The value of the C_{EXT} is fixed but the internal pull up resistance value varies depending upon the voltage supply of the MCU. $R_{\mathbf{PU}}$ decreases with the increase of $V_{\mathbf{DD}}$.

The variation of $R_{\mathbf{PU}}$ with $V_{\mathbf{DD}}$ is given in the datasheet.

7.2.2.1 Wait Mode

Methodology: An external CLK source, up to 16MHz is used on pin CLKIN. PLL is disabled and internal RC Oscillator is OFF. So, f_{OSC} is CLKIN / 2. All peripherals are in reset configuration except Ports. At reset the SMS bit is 0, hence f_{CPU} is f_{OSC}. The IS01 bit in the EICR register is cleared and the IS00 bit in the EICR register is set to 1. All Ports are in Push-Pull Output with low level except PA0 (which is used for charging and discharging the capacitor).

Measurements:

Table 18. Consumption I_{DD} (Wait Mode) for C_{EXT} = 1 uF at T_A = + 25 °C

External CLK on CLKIN Pin (MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
16	**8**	1850 µA	2087uA	2380uA
8	**4**	1195uA	1426uA	1590uA

7.2.2.2 Slow-Wait Mode

Methodology: An external CLK source, up to 16MHz is used on pin CLKIN. PLL is disabled and internal RC Oscillator is OFF. So, f_{OSC} is CLKIN / 2. All peripherals are in reset configuration except the I/O Ports and the SMS bit of in the MCCSR register is set to 1, hence f_{CPU} is f_{OSC}/32.The IS01 bit in the EICR register is cleared and the IS00 bit in the EICR register is set to 1. All Ports are in Push-Pull Outputs with low level except PA0 (which is used for charging and discharging the capacitor).

Measurements:

Table 19. Consumption I_{DD} (Slow-Wait Mode) for C_{EXT} = 1 uF at T_A = + 25 °C

External CLK on CLKIN Pin (MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
16	**0.25**	430uA	526uA	612uA
8	**0.125**	293uA	380uA	434uA
4	**0.0625**	220uA	280uA	333uA

7.2.2.3 Halt Mode

Methodology: An external CLK source, up to 16MHz is used on pin CLKIN. PLL is disabled and internal RC Oscillator is OFF. So, f_{OSC} is CLKIN / 2. All peripherals are in reset configuration except Ports. IS01 bit of EICR is cleared and IS00 bit of EICR is set to 1. All Ports are in Push-Pull Outputs with low level except PA0 (which is used for charging and discharging the capacitor).

Measurements:

Table 20. Consumption I_{DD} (Halt Mode) for C_{EXT} = 1 uF at T_A = + 25 °C

f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
8	95 µA	115 µA	143 µA

7.2.3 Conclusion

If V_{DD} increases, R_{PU} decreases. So, the RC time constant also decreases for a fixed value of C_{EXT}. So, the wake-up time decreases. Hence wake-up frequency increases with the increase of V_{DD}.

Temporization period exists only in Halt mode in case of Interrupt, because during all other modes (except Halt mode) oscillator is ON. So, in HALT mode, the MCU needs 256 CPU cycles as temporization period, during that time it consumes some current.

* Wake-Up frequency is the external interrupt frequency (Wake-Up time is the time after which the MCU is woken up from the different modes (Wait, Slow-Wait, Halt)).

7.3 EXAMPLE 3: APPLICATION WITH PERIODIC WAKEUP AND WATCHDOG

For this example, we assume that for the application the Watchdog Timer (WDG) has to remain active and that the microcontroller has to stay in low power mode for an undefined time period. Therefore, Halt mode can not be used, as the “Halt” instruction resets the microcontroller when the WDG is active and WDGHALT option bit in the option byte is set to 1. When the WDGHALT option bit in the option byte is 0 and WDG is active, a “Halt” instruction doesn’t generate a Watchdog reset. In this case, the MCU enters Halt mode. The Watchdog counter is decremented once and then stops counting and is no longer able to generate a watchdog reset until the MCU receives an external interrupt or reset. Active-Halt mode can also not be used, because, the device cannot spend more than a defined period in this power saving mode. Consequently, Wait mode is used, associated with Slow mode configured for the slowest frequency. This slow mode clock (f_{CPU}) drives the active peripherals and the CPU when it is woken-up. Of course, to reduce power consumption further, Halt mode should be used wherever possible.

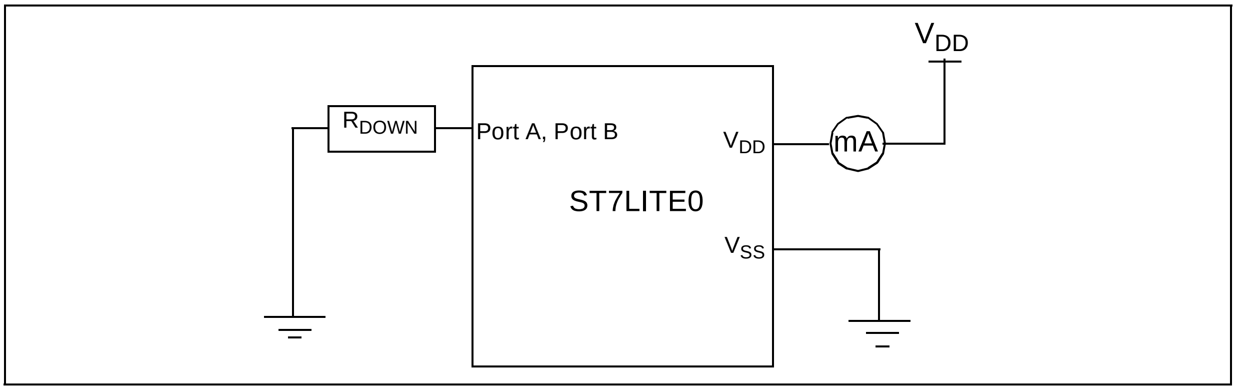
In terms of applications, users usually try to optimize the power consumption during the wait loop in the main program. In this configuration the microcontroller is mostly idle, waiting for an external event, or regularly woken-up by an internal timer event. In our case, we decided to put the ST7 core in Wait For Interrupt state, with one timer active to regularly wake up the core. This enables the core to refresh the WDG to prevent a watchdog reset. The clock driving the active peripherals and the core when it is active corresponds to the slowest clock available. It is the oscillator clock divided by 2 then divided by 32. So the clock driving the peripherals and core (f_{CPU}) is equal to the $f_{OSC}/32$. The measurement is done using the following configuration.

7.3.1 Measurement Configuration

In this mode, the MCU is configured as below, but of course in the application this configuration may differ, in particular the I/O port used.

– Slow mode is selected with the divided external clock factor equal to 64. This clock (f_{CPU}) drives the active peripherals (WDG and LT-Realtime Clock) and the core when it is active. In the Main Clock Control/Status Register (MCCSR) the SMS bit must be set.
– All I/O ports are connected to an external pull-up or pull-down to avoid leakage due to floating inputs.
– The Watchdog Timer (WDG) is active with a 2msec time-out period.
– Here 8 bit LT is used to wake-up the core at regular intervals, 1msec, to avoid a WDG reset.
– In LTCSR register set, TB to 0 & TBIE to 1.
– The main loop contains a Wait For Interrupt “WFI” instruction.
– Here all other peripherals (except above) are OFF
– Set Option Byte “value” to “FC FE” (In Option Byte, PLL = disabled, RC Oscillator = OFF, LVD = OFF, WDG SW = Software, WDG HALT= No Reset when Halt)
– For the ST7 software configuration, please refer to your datasheet.

Figure 6. Hardware Setup



7.3.2 Application

Methodology: An external CLK source, up to 16MHz is used on pin CLKIN. PLL is disabled and internal RC Oscillator is OFF. So, f_{OSC} is CLKIN / 2. All peripherals are in reset configuration and the SMS bit in the MCCSR is set to 1.Hence f_{CPU} is f_{OSC}/32. All I/O ports are connected to an external pull-down.

Measurements:

The measurement of current (I_{DD}) is done directly on the V_{DD} pin. The microcontroller is in Wait mode and is configured as described above, with the WDG and LT Timer active.
The values given in Table21 correspond to typical measured values. They are not maximum or minimum values.

Table 21. Consumption I_{DD} at T_A = + 25 °C

External CLK on CLKIN Pin (MHz)	f_{CPU}(MHz)	I_{DD} at V_{DD} = 4.5 V	I_{DD} at V_{DD} = 5 V	I_{DD} at V_{DD} = 5.5 V
16	**0.25**	369 µA	425 µA	500 µA
8	**0.125**	250 µA	283 µA	338 µA
2	**0.03125**	149 µA	187 µA	245 µA

7.3.3 Conclusion

Current consumption depends on V_{DD}. Current consumption increases with the increase of V_{DD} and F_{OSC}.