Jacob Honer

A summary of the NXP LPC1125 Microcontroller (MC) regarding GPIO function and Timers, compared to the NXP KL25Z128 MC, from NXP Semiconductors

ECE 331 Research Project 2

November 21st, 2020

Introduction

This paper will be an analysis of the GPIO and timer functionality of the NXP LPC1125 microcontroller, as well as offer insightful comparisons of the NXP LPC1125 MC and the NXP KL25Z128 MC regarding GPIO and timer functionality, in addition to their effectiveness in the design of a wearable health monitoring system. This paper will talk about the configuration and specifications of the NXP LPC1125 MC, in conjunction with an overview of the configuration registers required to set up the NXP LPC1125 for various GPIO functions. It will also address the different timers of the NXP LPC1125 and how to configure their respective configuration registers, as well as introduce the basics of the various timer registers that can be useful in programming. The paper will conclude with a comparison of how these timers vary between controllers and what that would mean for applications sake of the two different MCs discussed. All information for this report was gathered from the NXP LPC1125 and the NXP KL25Z128 User Manuals, taken directly from the NXP website [1], [2].

General Purpose I/O (GPIO)

Overview of functionality

The NXP LPC1125 comes equipped with 38 GPIO pins across four ports, port 0 through 3. Port 0 and 1 contains 12 GPIO pins, port 2 contains 10 GPIO pins, and port 3 contains only 4 GPIO pins. There are 10 other non GPIO pins on the MC, which include reset, voltage, ground, etc. Each GPIO pin comes with configurable pull-up/pull-down resistors and all GPIO pins are bidirectional. A configurable open-drain mode is also supported for some pins. All GPIO pins on this MC can be used as edge and level sensitive interrupt sources. This MC also includes a high-current output driver of 20 mA on one pin, as well as two high-current sink drivers of 20 mA on two of the I2C-bus pins in Fast-mode Plus, which despite not being a GPIO feature, is still good to know. Lastly, the analog pins of this MC include four general purpose counter/timers with up to six capture inputs and up to 13 match outputs, which can be useful if multiple timers are required for a certain task, powered by a Programmable Windowed Watchdog Timer (WDT).

Details of operation

Back to the GPIO pins and their functionality, the electrical characteristics of the I/O pins are configurable via the configuration registers, and these registers can set: the features for pin function, the settings for internal pull-up/pull-down resistor or bus keeper function, hysteresis, analog input or digital mode for pads hosting the ADC inputs, I2C mode for pads hosting the I2C-bus function, and a pseudo open-drain mode for non-I2C pins. For this analysis, we will mostly focus on the functionality of said registers with respect to their use as I/O pins, however, we will also explore some of the other features later on in this paper.

This Microcontroller (MC) uses I/O configuration registers, referred to also as IOCON registers in the manual, to set the settings for each MC pin on the four respective ports. The addressing of said registers will be discussed later on. Each IOCON register controls the PIO port pins, the inputs and outputs of all peripherals and functional blocks, the I2C-bus pins, and the ADC input pins. Juxtaposing this MCs methods for setting pin control with those of the NXP KL25Z128 studied in class, the IOCON registers would be equivalent to the Pin Control Registers (PCR) of the NXP KL25Z128 MC.

Regarding GPIO, the focus of this discussion, to set a pin to GPIO on the NXP LPC1125, the FUNC bits in the pins respective IOCON register must be set to 0x0. These three bits act as a selection bit of a 3-bit MUX, where the first selection is always GPIO; this mirrors the NXP KL25Z128 MC from class. For the NXP LPC1125 MC, the function bits are 2:0 for all pins that are GPIO configurable, which makes register configuration easy. It is also notable to know that for this MC, registers have only 2-5 functions to select from, with there being some diversity of function selection; regardless, these 3 bits are sometimes more bits than necessary, but for simplicity's sake, all registers are obviously kept the same size. For comparison's sake of the NXP LPC1125 and the NXP KL25Z128, the NXP KL25Z128 has also a 3-bit selection MUX with comparable functions, however, the NXP KL25Z128 uses bits 10:8 of each register, compared to bits 2:0 for the NXP LPC1125. Ultimately, this does not matter too much in terms of functionality, however, it is good to know and highlights why it is necessary to configure our compilers to each specific MC, as many settings, such as pin control register configurations, are not consistent across MCs.

For the GPIO pins of the NXP LPC1125, the GPIO DIR registers determine if a pin is an input or output pin. For the GPIO DIR registers, there are a total of four, one for each port, where bits 11:0 on each register correspond with the same numbered pin on the respective port. For these registers, a bit value of 0 indicates that a specific pin is configured as an input and a bit value of 1 would indicate that a pin is configured as an output. It is also important to note that these GPIO DIR registers only apply when a pin is configured for GPIO; for peripheral functions, the pin's direction is controlled automatically depending on the functionality of said peripheral function, hence, the GPIO DIR registers have no effect.

The MODE bits on each IOCON register sets the mode of the on-chip pull-up or pull-down resistors for each pin, or it selects the repeater mode function, to be discussed shortly. The possible on-chip resistor configurations are pull-up enabled, pull-down enabled, no pull-up/pull-down enabled, or the repeater mode. The MODE selection bits, which consist of bits 4:3 on each IOCON register, selects which resistor pull-up/pull-down configuration to be used for each pin. For the MODE selection bits of the IOCON registers, 0x0 configures to inactive or no pull-down/pull-up resistor enabled, 0x1 configures the pull-down resistor to be enabled, 0x2 configures the pull-up resistor to be enabled, and lastly, 0x3 configures repeater mode to be enabled. The repeater mode enables the pull-up resistor if the pin is set to HIGH logic and enables the pull-down resistor if the pin is set to logic LOW. This causes a pin to retain its last known state when it is configured as an input and is not driven externally. The reason that this can be useful is to prevent a particular pin from floating, which in turn prevents an individual pin from potentially using significant power, something that can happen when a pin floats to an indeterminate state. The standard I/O pin configuration, equipped with the pull-down/pull-up resistor setting can be found in figure 1 and can be referenced to aid in understanding of the I/O configuration of this MC.

The default value for the NXP LPC1125 is pull-up enabled, meaning that the pull-up enabled configuration would be the default status on RESET, for example. If the pull-up resistor is enabled, all non-I2C pins, which obviously includes the GPIO pins, are pulled up to 3.3 volts. This 3.3 volts is stems from the VDD pin being set to this 3.3-volt value.

The IOCON registers also include bits for enabling and disabling other configurations of non-GPIO functions on each pin, which apply when the FUNC bits of the IOCON register is not set to 0x0, but since this paper will focus on strictly the GPIO functionality of the NXP LPC1125, these other fits and their corresponding configuration functionalities will not be discussed.

The last thing to be discussed regarding the GPIO functionality of the NXP LPC1125 MC is the addressing of the discussed registers. Starting with the base address for each port, we have 0x5000.0000, 0x5000.1000, 0x5000.2000, 0x5000.3000 respectively for ports 0, 1, 2, and 3. The GPIO DIR registers are accessible from the base address for the respective port offset 0x8000. Not yet discussed, the DATA registers for each port can be found with an offset of 0x3FFC from the base addresses previously listed. It is important to note that the GPIO DIR registers and the DATA registers of the NXP LPC1125 would be equivalent to the PDOR and PDIR, respectively, of the NXP KL25Z128, and both work in identical ways when it comes to the PDOR or GPIO DIR registers effect on the PDIR or DATA registers.

The IOCON registers can be found with a base address of 0x4004.4000 plus their corresponding register specific offset. Unlike the PCR registers of the NXP KL25Z128, the IOCON registers of the NXP LPC1125 do not seem to follow a logical port based incremental addressing, which means that the programmer would have to do more work in organizing the addresses in code. Ultimately, this is not too much of a problem though, as each pins IOCON register address can be easily found in the tables of the NXP LPC1125 Reference Manual, and once these values are written in code, they can be easily stored and accessed via structs or another organizational element of the C programming language.

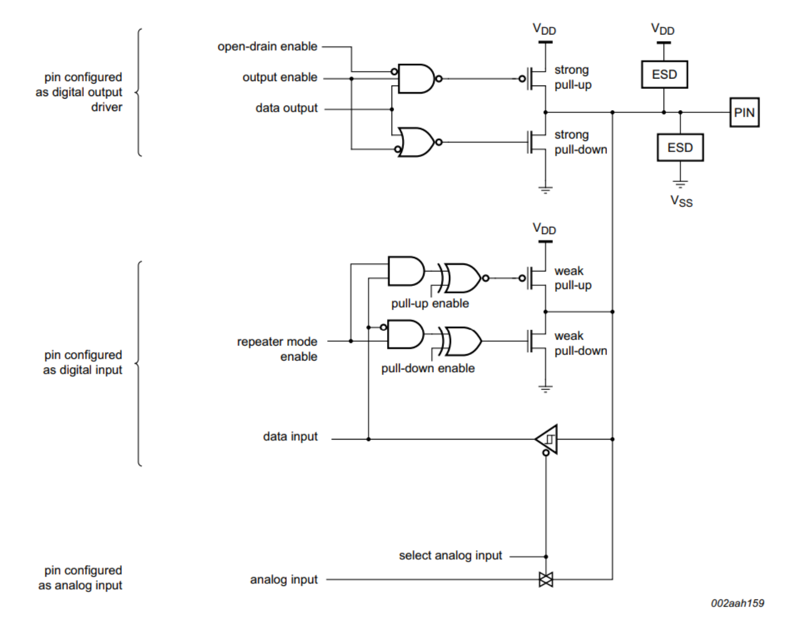

Figure 1 – The standard I/O pin configuration of the NXP LPC1125 MC [1].

Comparison of the NXP LPC1125 to the NXP KL25Z128

Regarding GPIO functionality of the NXP LPC1125 in comparison to the NXP KL25Z128, their functionalities are quite similar. As mentioned, both have some differences regarding the names of their registers, the organization of their pin configuration registers, and the registers locations in memory, but ultimately these are minor differences to be expected when comparing different MCs. Both MCs also allow for pull-up/pull-down resistor configuration via their respective pin configuration registers. A small but interesting difference regarding these configuration registers is that to select GPIO pin function on the NXP LPC1125, the FUNC bits of the IOCON register are set to 0x0, whereas to select the GPIO pin function on the NXP KL25Z128, the Pin Mux Control bits of the PCR is set to 0x1.

One of the larger differences, however, between the two MCs concerning GPIO is the number of pins that each controller has that is GPIO configurable. The NXP LPC1125 has up to 38 GPIO pins, whereas the NXP KL25Z128 has up to 66 GPIO configurable pins, which can be a rather drastic difference as far as applications go.

With all things considered, despite the discrepancy in the number of pins of each MC, there are no major differences between the NXP LPC1125 and NXP KL25Z128 regarding GPIO functionality, although small design differences on each MC might lead to a learning curve if transitioning from one MC to the other.

Timers and timer module

Overview of functionality

The NXP LPC1125 comes equipped with a System Tick Timer (SysTick timer), as well as four general purpose counters/timers with up to 6 capture inputs and up to 13 match outputs; of the four general purpose counters/timers, two are 32-bit general-purpose counter/times, and two are 16-bit general purpose counter/timers, all of which are individually configurable to be discussed.

Before discussing Timers, it is important to discuss clock generations of the MC. The Clock Generation Unit of the NXP LPC1125 has three independent oscillators, the system oscillator, the Internal RC oscillator (IRC), and the watchdog oscillator. Following reset, the MC always restarts using the IRC oscialltror until it is switched via software; this is so that the MC by default is operating at a consistent known frequency. The IRC operates at a frequency of 12 MHz, the system oscillator has an operating range of 1 MHz to 25 MHz, and the watchdog oscillator has an operating frequency range of 600 kHz and 4.6 MHz. The operating range of each oscillator is configurable via their respective control registers. An important thing to note is that the watchdog oscillator operates at a ±40% accuracy from the listed frequency value, meaning it is not very accurate compared to the ICR which operated at ±1%, however, it is the lowest power consumption clock on the MC. The watchdog oscillator also must be programmed to by writing to its respective control register before it is usable. Furthermore, the control register of the system oscillator, notably, allows the programmer to set the operating frequency range when operating in low power modes, and is also in control of bypassing any external clock source that is connected to the MC via an external oscillator.

The respective oscillators can be modified and configured via the system PLL, as well as an array of dividers. Each PLL or divider has its own respective control register that controls the processes of said PLL or divider. All dividers divide by a value of 1 through 255, and the PLL accepts an input frequency from 10 MHz to 25 MHz, which is multiplied up to a high frequency, then divided down given the control register specifications. It is not necessary to go into details regarding the PLL or each divider, as it is rather straightforward, but it is important to note that the system clock that is generated from this process is used by the SysTick timer, as well as the four other general-purpose counters/timers of this MC. The LPC112x User Manual gives a concise explanation of each register and its purpose, so it will not be further discussed here, but the table in figure 2, which comes from the manual, gives a detailed breakdown of the clock generation and connections.

SysTick timer details of operation

The SysTick timer will first be discussed. The SysTick timer is a 24-bit timer that uses a dedicated exception vector for system exceptions and is clocked internally by either the system clock or the system clock/2, depending on how the CLKSOURCE bit of SysTick Timer Control and status register is configured. The SysTick timer is an important part of the ARM Cortex-M0 architecture and it intended to generate a fixed 10 millisecond interrupt for the system. The SysTick timer makes use of only 4 unique registers. The System Timer Control and status register is one of these registers, which as mentioned earlier, is used to enable the timer, enable system interrupts, set the clock source, as well as set the COUNTFLAG bit, which is a 1 if the SysTick timer counted to 0 since the last read of the register, otherwise 0. Second, we have the System Timer Reload value register which holds the 24-bit value that is loaded into the SysTick timer whenever it counts down to zero. Next, we have the System Timer Current value register that, almost obviously, returns the current value of the SysTick counter. Lastly, there is the System Timer Calibration value register that holds calibrations values for the registers. This SysTick can be used really whenever a general-purpose timer is desired in the program, as well as for certain operating system or system management specific processes.

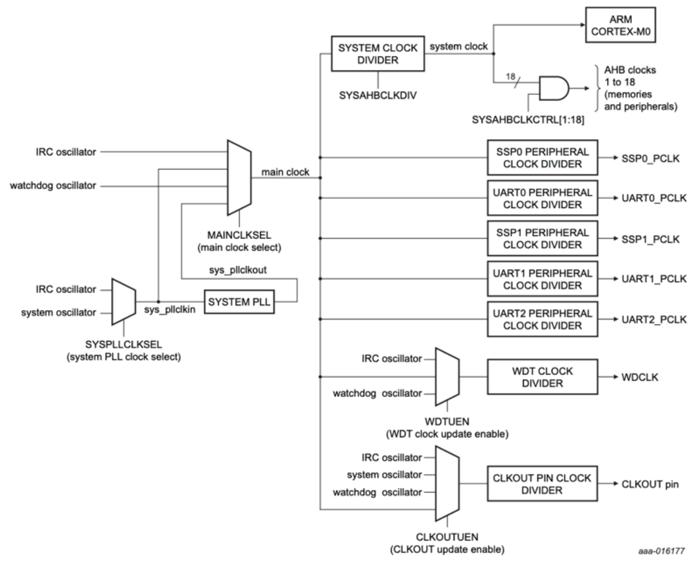

Figure 2 – The clock generation schematic of the NXP LPC1125 with each divider and PLL shown [1].

16-bit general purpose counter/timer details of operation

The next timer that this paper will explore is the 16-bit general purpose counter/timer that the NXP LPC1125 also offers. This timer is slightly more complex than the SysTick discussed above. Each 16-bit general purpose counter/timer can be used for counter or timer operations, but also it can be configured to be cleared on a specific capture event or to generate an interrupt. This means that it is easy to measure a pulse-width by clearing on the leading edge of an input and capturing on the trailing edge. Given this pulse width moderation (PWM), this timer would provide itself to be rather useful when considering the design of a wearable health monitoring application. These two counter/timers also come with four 16-bit match registers that can be used for: continuous operation, stopping operation, or settings the timer on match, with an optional interrupt on each. They also have up to three output registers that can be set to high, low, toggled, or do nothing on match.

To configure this Timer, first, the respective CT16B pin of the IOCON register must be set to 1 to enable the Timer. The respective enabler pins of the SYSAHBCLKCTRL register, which gates the system clock to the various peripherals and memories, must also be appropriately set. Each timer comes with an array of registers that are used to configure the timer's functionality. To understand the functionality of this timer, the timer's registers will be discussed. Starting out is the interrupt register that holds the interrupt flags for the match register interrupts and or capture register interrupts that are set when a match occurs. Next is the timer control register which enables the timer and the timer counter register that holds the counter value. The prescale register sets the maximum value for the prescale counter register, and this value controls the resolution of the timer counter, as this register increments every clock cycle, and only once this register reaches its maximum value does the timer counter actually increment. Following is the match control register that specifies what operation to be performed when one of the match registers matches the timer counter - the options for each match controls by the match registers follows interrupt, reset, and stop. Then comes the 4 unique match registers that hold the match value. Performing a similar process as the match control register is the capture control register that enables capture on the rising and/or falling edge, as well as enabling an interrupt on match. The capture registers, almost obviously, hold the values of its associated device pin. We then have the external match register, and this register provides both control and status of the external match channels and external match pins and is the register that would probably be used in the code for most clock related functions that is unique from the SysTick timer discussed earlier. Lastly, we have the count control register that sets the settings of counter verse timer mode, as well as what input to select and if it is a rising or falling edge of the capture register that clears the timer, and the PWM register that enables and disables PWM from to the four different channels.

This timer might seem complex but is really quite simple once each register is understood. Another useful tool to understand the functional use of this timer is the 16-bit counter/timer block diagram that can be found in figure 3. In short, this timer is a great tool for not only counter/timer related processes, but also PWM, and matching, and this is powerful by its capture and match registers, as well as it being rising, falling, or both edge sensitive. All four of general-purpose counter/timers of this MC function similarly to the one outlined above.

Comparison of Timer functionality of the NXP LPC1125 to the NXP KL25Z128

The following will be a brief comparison of the possible Timers of the NXP LPC1125 and the NXP KL25Z128. As discussed, the NXP LPC1125 has a SysTick Timer and two general purpose 16-bit counter/timers and two general purpose 32-bit counter/timers. The NXP KL25Z128, however, also has a SysTick Timer, but also a 2-channel periodic interrupt timer, a low-power timer, a real time clock, and then three PWM timer modules, one that is 6-channel PWM timer and two that are 2-channel PWM timers. Comparing the two, the NXP KL25Z128 definitely wins in terms of the number of different timers and the different timers for different tasks, as it has a 2-channel periodic interrupt timer, a low-power timer, and a real time clock, all timers that the NXP LPC1125 lacks. The NXP LPC1125 Timers can perform the same functions as the 3, however, but the process of configuration would be less straightforward and might require some round about solutions that the three timers of the NXP KL25Z128 would easily handle. Regarding the PWM timers, however, the NXP LPC1125 does have one more than the NXP KL25Z128 and the functionality of both PWM configurable timers seem to be identical. A notable difference between the PWM configurable timers of the NXP LPC1125 is that two of them are 32-bit timers, whereas the PWM timers of the NXP KL25Z128 are all 16-bit timers. This could be a limitation if the modulation occurs outside of the clock range that the timer can perform at, but for most cases would not be a problem. With respect to the design of a wearable health monitoring application, despite the NXP KL25Z128 having more GPIO pins and more timer options, if this 32-bit timer is super beneficial, this would give a major edge to the NXP LPC1125 that, thus far, seems to be lagging behind the NXP KL25Z128 in terms of functionality for an array of different tasks.

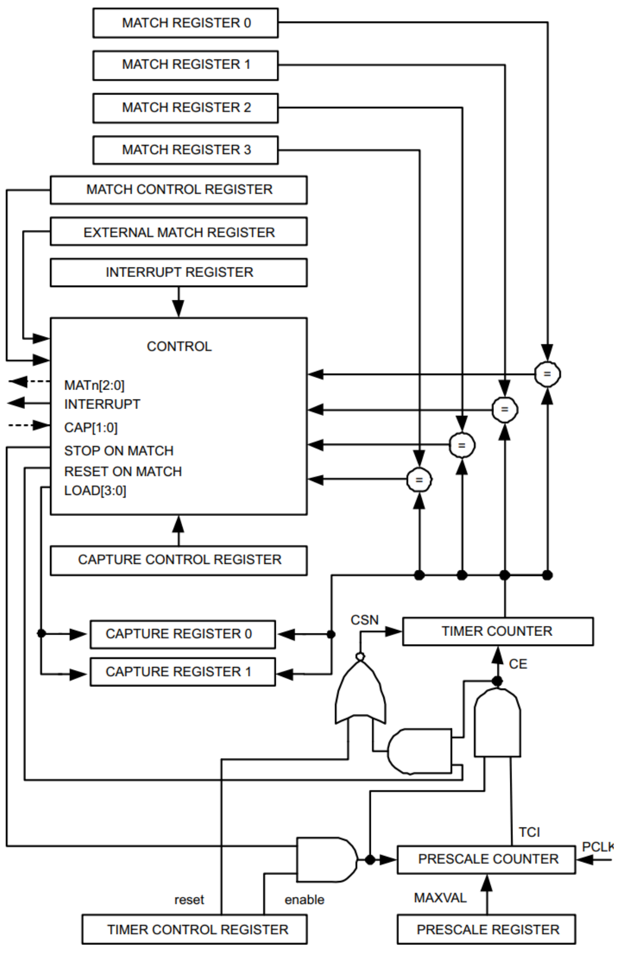

Figure 3 – The timer schematic for the general purpose 16-bit counter/timers [1].

References:

[1] Nxp.com. 2020. *Lpcxpresso1125 Board | NXP*. [online] Available at: <https://www.nxp.com/design/microcontrollers-developer-resources/lpc-microcontroller-utilities/lpcxpresso1125-board:OM13080> [Accessed 25 October 2020].

[2] Nxp.com. 2020. *FRDM-KL25Z|Freedom Development Platform|Kinetis® MCU | NXP*. [online] Available at: <https://www.nxp.com/design/development-boards/freedom-development-boards/mcu-boards/freedom-development-platform-for-kinetis-kl14-kl15-kl24-kl25-mcus:FRDM-KL25Z> [Accessed 25 October 2020].

** all information regarding the NXP LPC1125 were obtained from the first citation, and all information pertaining to the NXP KL25Z128 were obtained from the second citation. Both are citations for each MCs respective manuals obtained from the NXP website.