Seven Segment API

York University

Dept. of Computer Science and Engineering

Joshua Abraham

Table of Contents

YouTube Demonstration: https://youtu.be/poy3gHdcU70

Overview

The seven segment API allows users to easily setup the software for their 4 digit seven-segment display on the LPC802. The library allows users to display numerical digits 0-9, and augmented versions of the alphabet A-Z, and a subset of special characters.

1. The following header file must be placed within the user's application:
   #include "seven_segment.h"

2. First pin assignments need to be passed into the API, to inform the program as to which GPIO pins will be used. This is used with the ***digitGPIOSetup*** and the ***sevenSegmentGPIOSetup*** with passed lists of the GPIO pins used for the 4 digits, and 7 segments respectively. The decimal place GPIO reference can be set independently using ***enableDecimalSegment***. Users may also use the ***sevenSegmentFullSetup*** which includes digits, the 7 segments, and the 8th segment for the decimal instead of the three aforementioned functions.

3. Once the display is configured, the user can choose between a wide assortment of display options including: single character display, single number display, four digit display, four character display, counter display, carousel display of any size, or a slider display of any size.

4. If a user implements an option that requires timers, there is the option of the following counters for use: SysTick, WakeUp timer, MultiRate timer (both channels), and the CTimer (channel 0). This allows users to pick the timers with flexibility to better fit their project needs. The API will automatically configure the clock specified (based off the clock the user must set themselves).

5. The interrupts, however, need to be generated by the user, but they simply need to add the respective API call in the interrupt. For example, if MRT0 is used to display 4 digits using the ***display4Numbers*** function, the ***display4NumbersInterrupt*** should be placed in the MRT0 interrupt service routine. This allows users to add additional logic to the interrupt if their system requires multiple actions to take place when the interrupt is fired.

6. Once the display is running the desired message, there are additional functions which an interact with the display for displays that have timers involved.

API Contents:

1.) A header file including prototypes for all user available functions, as well as an array including all accepted characters
2.) A c file containing the source code for the header file
3.) An example file to demonstrate setup and use

API Functions

Setup Functions

This portion of the API is designed to allow fast integration of a seven-segment display on various sets of pins increasing flexibility. It also is robust in the subsystems present, allowing for specific control over both common-anode and common-cathode displays, as well as optional 8th segments. This subsystem is integrated into the API, allowing the use of a wider array of displays without major modifications needed.

digitGPIOSetup

Description: Sets up the GPIO assignments for the 4 GPIOs to be used for digits.

Parameters:

- *channels*: list of up to 4 GPIO pins to be used for the digits of the display

sevenSegmentGPIOSetup

Description: Sets up the GPIO Assignments for the 4 GPIOs to be used for digits.

Parameters:

- *segs*: list of up to 7 GPIO pins to be used for the display (Should be ordered from Segment A … Segment G)

setSevenSegmentType

Description: Sets up the type of seven segment to be used (common anode or cathode).

Parameters:

- *type*: 0 = common anode, otherwise common cathode

enableDecimalSegment

Description: Enable or disable the decimal segment.

Parameters:

- *decimalSegment*: GPIO Pin for decimal point, -1 means disabled

sevenSegmentFullSetup

Description: Do a full reset/setup rather than just portions of the display.

Parameters:

- *channels*: list of up to 4 GPIO pins to be used for the digits of the display

- *segs*: list of up to 7 GPIO pins to be used for the display (Should be ordered from Segment A … Segment G)
- *decimalSegment*: GPIO Pin for decimal point, -1 means disabled

toggleDecimalPoint
Description: Turn on the decimal point for the display if previously off, otherwise turn the decimal point on.

setDecimalPoint
Description: Turns on the decimal point segment.

clearDecimalPoint
Description: Turns of the decimal point segment.

Non-timer Based Functions
This portion of the API deals with sending a single character to all four digits of the display. This is a parallel interface as all four digits are controlled simultaneously.

displaySingleCharacter
Description: Display a single character on all 4 digit displays simultaneously, there is no need for a timer for this function. Example uses include displaying "0000", "8888" or "----" for example.

Parameters:

-*inputChar*: Single character to display on all active digit displays

displaySingleInt
Description: Display a single integer on all 4 digit displays simultaneously. For example, if a user wishes to display "0000" or "8888".

Parameters:

-*inputNum*: Single integer to display on all active digit displays

Timer Based Functions
This portion of the API deals with sending a set of character to all four digits of the display. This is a serial interface as all four digits are controlled sequentially to optimize the number of pins needed for the display. This system also involves further complexities in timing, within the slider and carousel animators. These systems involve two clocks that work in parallel, where the first controls cycling through the 4 visible digits, and the second deals with transitioning in new sets of 4 digits for the first timer to display.

display4Characters

Description: Function used to display 4 characters continuously on the seven-segment display. This function calls the timer configurations internally.

Parameters:

- *inputSequence*: Takes 4-character input
- *clockType*: Input options include "SysTick", "WKT", "MRT0", "MRT1", "CTIMER0"
- *refreshRate*: how fast the 4 characters are cycled through on the seven-segment display

display4Numbers

Description: Function used to display 4 numbers continuously on the seven-segment display.

Parameters:

- *inputNumber*: Takes a 4-digit input
- *clockType*: Input options include "SysTick", "WKT", "MRT0", "MRT1", "CTIMER0"
- *refreshRate*: how fast the 4 characters are cycled through on the seven-segment display

setupSevenSegmentCounter

Description: This function used to display a counter on the seven-segment display. This function calls the 2 timers' configurations internally. Calling the respective interrupts will refresh the display and update the counter respectively. Note that the *counterClock* and the *refreshClock* cannot be set to the same clock.

Parameters:

- *clockStart:* Starting time for the display
- -*clockType:* Input options include "SysTick", "WKT", "MRT0", "MRT1", "CTIMER0"
- *newCountDirection*: input character sequence "UP" or "DOWN"
- *newCountIncrement:* counter increment/decrement depending on direction (negatives are allowed)
- *newStopValue:* the early stopping point for counting, the value must be exactly hit not just surpassed
- *enableStopValue:* intput either true to enable the stop value or false otherwise
- *newCountRate:* Speed of counting progression
- *refreshClock:* Input options include "SysTick", "WKT", "MRT0", "MRT1", "CTIMER0"
- *refreshRate*: Input an integer to indicate how fast the numbers are cycled through on the seven-segment display

sevenSegmentDisplayTextCarousel

Description: This function is used to display a carousel of characters on the seven-segment display. This function calls the 2 timers' configurations internally. Calling the respective interrupts will refresh the display and update the transition of the carousel. The character sequence will loop through in a counter clockwise direction. Note that the *newTransisitionClock* and the *refreshClock* cannot be the same timer.

Parameters:

- characterSequence: As long of a sequence as desired of characters ("Hello World")
- *sequenceLength*: Length of Sequence
- *newTransitionClock*: Input options include "SysTick", "WKT", "MRT0", "MRT1", "CTIMER0"
- *transitionSpeed:* Speed of carousel motion
- *newEnableContinousCycle*: true implies the carousel will cycle continuously, false implies a one-shot action
- newEnablePadding: Pad the sequence with a blank start and end screen
- refreshClock: Input options include "SysTick", "WKT", "MRT0", "MRT1", "CTIMER0"
- refreshRate: speed of cycling through digits

sevenSegmentDisplayTextSlider

Description: Creating a slider for text to display 4 characters at a time and then switch to the next set of four characters. This function calls the 2 timers' configurations internally and calling their respective interrupts will cycle between the four displayed digits and transition to the next character sequence. Note that the *newTransisitionClock* and the *refreshClock* cannot be the same timer.

Parameters:

- *characterSequence*: As long of a sequence as desired of characters ("Hello World")
- *sequenceLength*: Length of Sequence
- *newTransitionClock:* Input options include "SysTick", "WKT", "MRT0", "MRT1", "CTIMER0"
- *transitionSpeed:* Speed of slider animation to cycle between character subsets
- *newEnableContinousCycle*: true implies the slider will cycle, false implies a one-shot action
- *newEnablePadding:* Pad the sequence with a blank start and end screen
- *ignoreSingleSpaces*: Input either true or false. Inputting true will ensure that sequences with single spaces have the single spaces ignored. Note that double spaces are not affected by this functionality. Example input and output of using the *IgnoreSingleSpaces* "A B C D E F G" will result in the following string being displayed "ABCD EF G"
- *refreshClock*: Input options include "SysTick", "WKT", "MRT0", "MRT1", "CTIMER0"
- *refreshRate*: speed of cycling through digits

Timer Interrupt Functions

display4CharactersInterrupt

Description: Used to swap between digits quickly on the display. This interrupt function should be called within the interrupt from the timer associated with the *refreshRate*. This function is used whenever ***display4Characters*** is used.

display4NumbersInterrupt

Description: This function should be used to swap between digits quickly on the display within the interrupt from the timer associated with the refresh rate. This function acts as a wrapper function for clarity, and ***display4CharactersInterrupt*** can be equally used whenever ***display4Numbers*** is utilized.

displayCarouselInterrupt

Description: This function should be used to swap between digits quickly on the display within the interrupt from the timer associated with the refresh rate. This function acts as a wrapper function for clarity, and ***display4CharactersInterrupt*** can be equally used whenever ***sevenSegmentDisplayTextCarousel*** is utilized.

displaySliderInterrupt

Description: This function should be used to swap between digits quickly on the display within the interrupt from the timer associated with the refresh rate. This function acts as a wrapper function for clarity, and ***display4CharactersInterrupt*** can be equally used whenever ***sevenSegmentDisplayTextSlider*** is utilized.

updateSevenSegmentCounterInterrupt

Description: This function should be used within the interrupt for the counter mechanism if the seven-segment display is used as a counter. This function will update the count for the seven-segment display. This function should be called within the interrupt that handles clock incrementing or decrementing. Use this interrupt function whenever ***setupSevenSegmentCounter***, where the clock associated with counting was passing in to *counterClock*.

sevenSegmentCarouselInterrupt

Description: This function should be nested within the interrupt for the carousel mechanism if the seven-segment display is used as a carousel. The function updates the currently displayed characters. This function should be called within the interrupt service routine associated with the *newTransitionClock*, whenever ***sevenSegmentDisplayTextCarousel*** is utilized.

sevenSegmentSliderInterrupt

Description: This function should be nested within the interrupt for the slider mechanism if the seven-segment display is used as a slider. The function updates the currently displayed characters. This function should be called within the interrupt service routine associated with the *newTransitionClock*, whenever ***sevenSegmentDisplayTextSilder*** is utilized.

Seven-Segment Counter Helper Functions

The helper functions listed in the next few sections are about small subsystems integrated into the larger system. These systems are very short functions with high impact, they give users real time control over functions that would be otherwise hidden from use.

getSevenSegmentDisplayCount

Description: Getter for the current normalized (displayed) count. Normalized implies the currently displayed count value (between 0000 and 9999). The actual count value may be larger as the counter wraps around when it hits the edge values. To get the actual count value use ***getSevenSegmentTotalCount*** instead.

Return:

- Integer *normalizedCount*

getSevenSegmentTotalCount

Description: Getter for the actual count value of the counter. If overflow within the counter has happened and the counter wraps around the display value may differ from the actual count. To get the normalized (currently displayed) count value use ***getSevenSegmentDisplayCount*** instead.

Return:

- Integer *currentCount*

toggleSevenSegmentCounterPause

Description: Toggle for the counter to pause or run based off previous mode.

pauseSevenSegmentCounter

Description: Pause the execution of the counter if previously running, if already paused there is no change.

runSevenSegmentCounter

Description: Run the execution of the counter if previously paused, if already running there is no change.

resetSevenSegmentCount

Description: This function resets the counter to the original value and will also restart the counting sequence if it has stopped.

setSevenSegmentCount

Description: This function will change the count value to a new integer value. It will update the displayed and normalized value and restart the counter. Doing this may cause a need to update the users *stopValue*.

Parameters:

- *newCount*: new count value to be displayed and stored

updateSevenSegmentIncrementer

Description: Change the count increment/decrement value to a new number, the counter will now change based off this new value.

Parameters:

- *newIncrement*: an integer value for changing the increment or decrement amount

changeSevenSegmentCountDirection

Description: Change the count direction to either "UP" or "DOWN". Note that if an incorrect string is entered, no change will be applied.

Parameters:

- *newDirection*: character sequence of either "UP' or "DOWN" as required

setCountStopValue

Description: Change the stop value for the counter. Doing this will also enable the stop condition automatically. It will also continue the counter if previously paused but will pause the counter if the new stop value equals the current count exactly.

Parameters:

- *newStopValue*: the new integer value for the counter to stop at

clearCountStopValue

Description: Clear the stop value for the counter. Doing so will also disable the stop condition automatically and continue the counter if previously paused.

Seven Segment Carousel Helper Functions

togglePauseSevenSegmentDisplayCarousel

Description: Toggle the pause on carousel motion if currently enabled, otherwise it will start motion.

pauseSevenSegmentDisplayCarousel

Description: Pause the carousel motion if currently moving, no change otherwise.

runSevenSegmentDisplayCarousel

Description: Continue the carousel motion if paused, otherwise no change in behaviour.

restartSevenSegmentDisplayCarousel

Description: Start carousel from beginning sequence again and will also remove the pause on the carousel if any exist.

Seven Segment Slider Helper Functions

pauseSevenSegmentDisplaySlider

Description: Pause slider animation for the display if currently moving, otherwise no change.

togglePauseSevenSegmentDisplaySlider

Description: Toggle whether the slider animation is running or not for the display.

runSevenSegmentDisplaySlider

Description: Run slider animation for the display if currently paused, otherwise no change in behaviour.

restartSevenSegmentDisplaySlider

Description: Restart slider animation for the display and doing this will also run the display if previously paused.