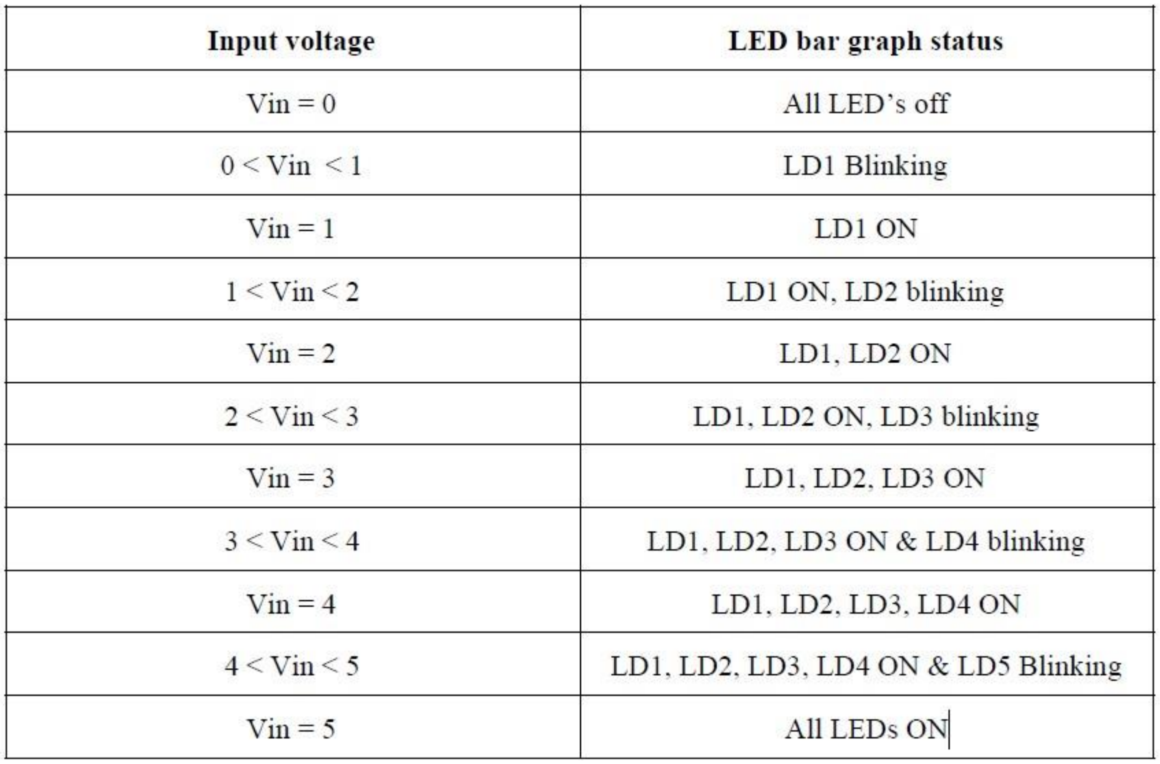
EMBEDDED SYSTEM DESIGN LAB ASSIGNMENT -2

STM8 Analog to digital converter peripheral programming and Interfacing

SUBMITTED BY- SANCHIT AGRAWAL (2016H140106)
ABHINAV GARG (2016H140103)

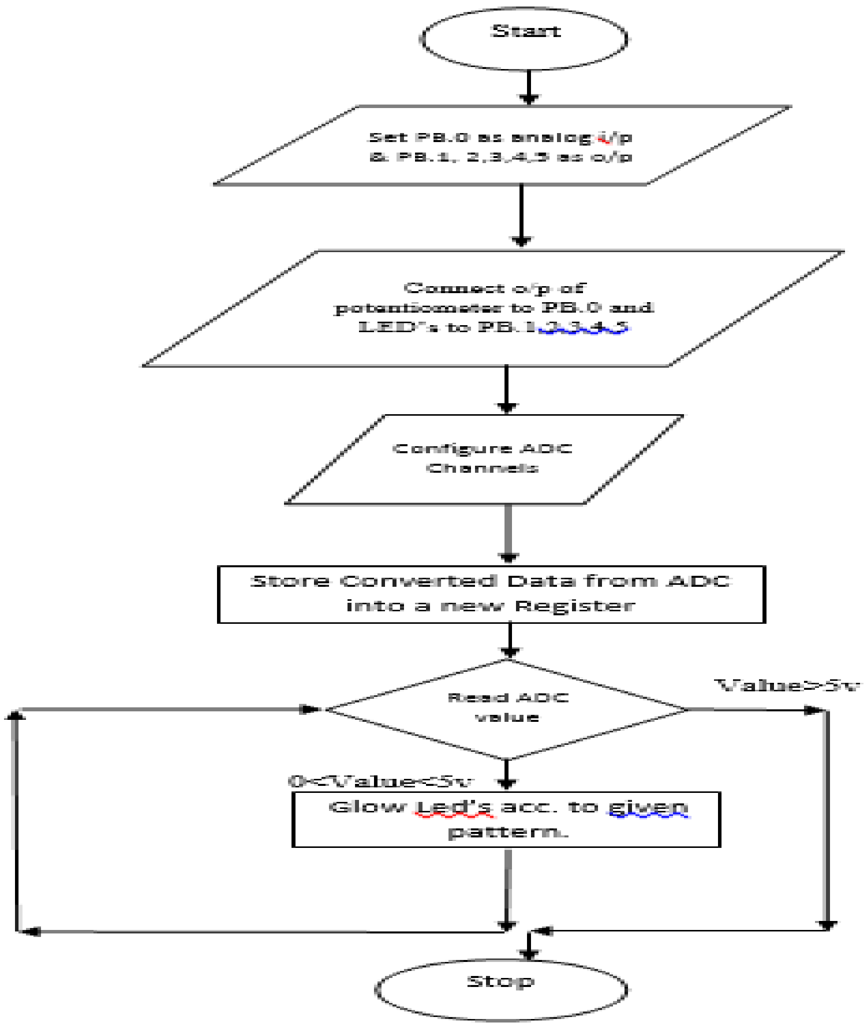
Objective – To Design a voltage measurement system using STM8 ADC peripheral. Measure the input DC voltage [0-5V] and display the measured voltage on LED bar graph. Input voltage to the system can be provided using a linear potentiometer. **Desired Led Pattern -**



Input voltage	LED bar graph status
Vin = 0	All LED's off
0 < Vin < 1	LD1 Blinking
Vin = 1	LD1 ON
1 < Vin < 2	LD1 ON, LD2 blinking
Vin = 2	LD1, LD2 ON
2 < Vin < 3	LD1, LD2 ON, LD3 blinking
Vin = 3	LD1, LD2, LD3 ON
3 < Vin < 4	LD1, LD2, LD3 ON & LD4 blinking
Vin = 4	LD1, LD2, LD3, LD4 ON
4 < Vin < 5	LD1, LD2, LD3, LD4 ON & LD5 Blinking
Vin = 5	All LEDs ON

Methodology – The voltage is varied using a Potentiometer and the variable voltage is provided to AIN 4. 5 Led are connected to pin3-pin6 of port A and pin0 of port G. The ADC is configured to provide data according to the voltage at AIN 4 of Port B. The result of ADC is stored in DR Registers. We have used Continuous mode of Conversion and Prescaled the Master frequency with reference to 3 different prescaling factors (Factor of 8 for low speed blinking, 6 for medium level, prescaling factor of 4 for high level).The LED blinking becomes fast as soon as the voltage appears near the set threshold values. The data converted after ADC is Right aligned.

FLOW CHART FOR ADC TESTING-


Start
Set PB.0 as analog i/p & PB.1, 2,3,4,5 as o/p
Connect o/p of potentiometer to PB.0 and LED's to PB.1,2,3,4,5
Configure ADC Channels
Store Converted Data from ADC into a new Register
Read ADC value
Value>5v
0<Value<5v
Glow Led's acc. to given pattern.
Stop

HARDWARE DESIGN –

Used 5 LED and connected them to Pin 3-Pin 6 of port A and Pin 0 of port G. Output resistors of value 220ohms are also used to limit the current. The LED are driven by the STM8S105C6 microcontroller and the input is received by AIN4.

The LED's are driven by the STM8S105C6 microcontroller and the input is received by potentiometer as shown in schematic diagram in next page.

SCHEMATIC FOR ADC TESTING-

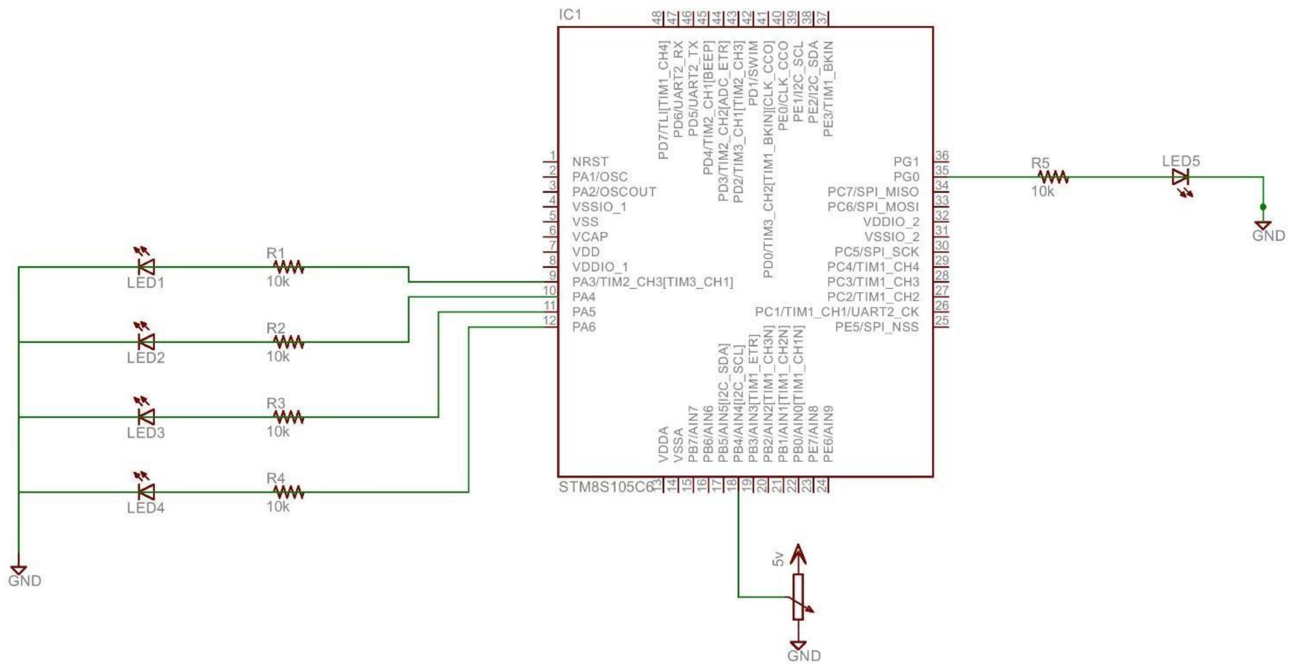


Figure 2: Schematic Diagram for ADC Testing

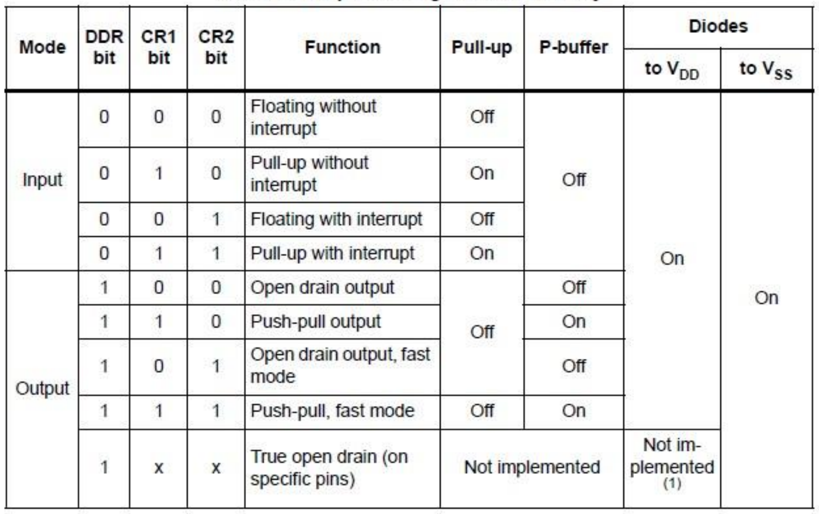
SOFTWARE DESIGN-

Used General Purpose Input Output (GPIO) function to control the input and output i.e. LED’s by making use of Data Direction Register (DDR), Input Data Register (IDR), Output Data Register (ODR), and Control Registers 1 (CR1 & CR2)

EXPLANATION-

1. The Header file (iostm8S105C6.h) includes all the input-output functions. 2. Header file “intrinsics.h” includes intrinsic function definition such as __disable_interrupt(), __enable_interrupt (here “__” represents an intrinsic function).
3. Then, structure named PD_ODR_bit is used, structure must be defined in the header file to manipulate each bit of ODR register.
4. By defining the above structure we set all the LED’s.
5. Global integer “int” and “ldnum” is defined and initialized with count=0. These will be used in both main function and interrupt handler function.
6. The “**pragma**” directives control the behaviour of the compiler, for example how it allocates memory for variables and functions, whether it allows extended keyword, and whether it outputs warning messeges. The pragma directives are always enabled in the compiler.
7. **"__interrupt"** is a **function attribute**, it adds an attribute to the function that it neither receive nor pass any argument.
8. In the main program loop, **“__disable_interrupt()”** is used. The significance of “__” is **intrinsic function** calling (from intrinsic.h file) for fast execution.
9. The various function performed on setting the DDR, IDR, CR registers at various pin positions is shown in table below-

Table 21. I/O port configuration summary

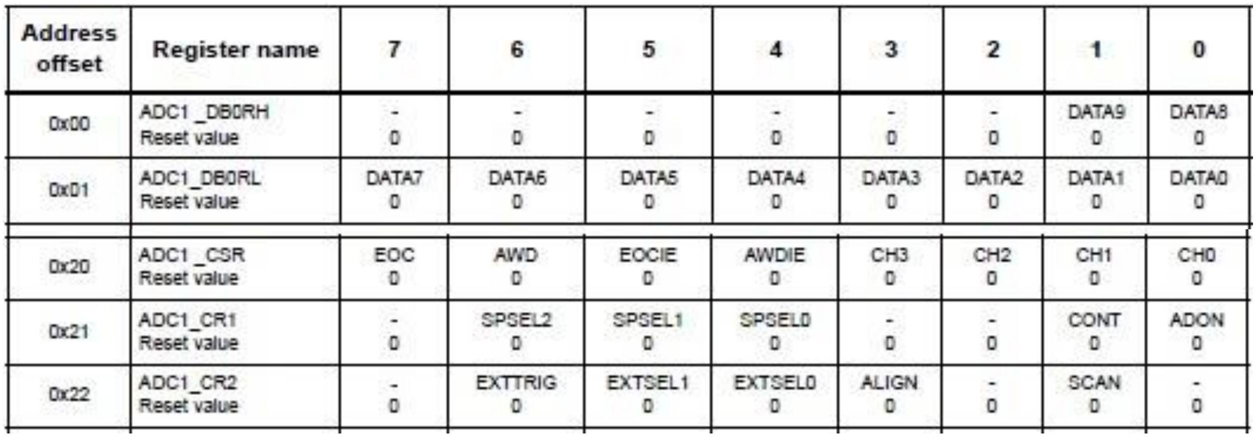


Mode	DDR bit	CR1 bit	CR2 bit	Function	Pull-up	P-buffer	Diodes	
							to V_{DD}	to V_{SS}
Input	0	0	0	Floating without interrupt	Off	Off	On	On
	0	1	0	Pull-up without interrupt	On			
	0	0	1	Floating with interrupt	Off			
	0	1	1	Pull-up with interrupt	On			
Output	1	0	0	Open drain output	Off	Off		
	1	1	0	Push-pull output		On		
	1	0	1	Open drain output, fast mode		Off		
	1	1	1	Push-pull, fast mode	Off	On		
	1	x	x	True open drain (on specific pins)	Not implemented		Not implemented (1)	

1. The diode connected to V_{DD} is not implemented in true open drain pads. A local protection between the pad and V_{OL} is implemented to protect the device against positive stress.

Figure 3: I/O port Register Mapping

10. Next, we set our output ports where the LED’s are connected
11. CR 1 register is initialized which is responsible for push pull operation.
12. **PD_ODR_bit structure** is used. In general structure must be defined in header file to manipulate each bit of ODR register.
13. Then the switch case operations for ADC working (as required for our operation) are set with the help of if-else ststements.The operation executes as soon as the interrupt arrives.



Address offset	Register name	7	6	5	4	3	2	1	0
0x00	ADC1 _DB0RH Reset value	- 0	- 0	- 0	- 0	- 0	- 0	DATA9 0	DATA8 0
0x01	ADC1_DB0RL Reset value	DATA7 0	DATA6 0	DATA5 0	DATA4 0	DATA3 0	DATA2 0	DATA1 0	DATA0 0
0x20	ADC1 _CSR Reset value	EOC 0	AWD 0	EOCIE 0	AWDIE 0	CH3 0	CH2 0	CH1 0	CH0 0
0x21	ADC1_CR1 Reset value	- 0	SPSEL2 0	SPSEL1 0	SPSEL0 0	- 0	- 0	CONT 0	ADON 0
0x22	ADC1_CR2 Reset value	- 0	EXTTRIG 0	EXTSEL1 0	EXTSEL0 0	ALIGN 0	- 0	SCAN 0	- 0

Figure 4: ADC Registers mapping

14. Here internal interrupt timer is used for various prescaling factors which is responsible for different LED blinking speeds.
15. All the timer register initialisations are done, and “CEN” of CR1 register is set to start the timer.
16. Timer count varies from 0 to 65535.
17. Then a function with function name “ldnum” is defined, this is responsible for the blinking of LED.
18. The zeroth bit (UIF) of SR1 register of TIM2 is set so as to update interrupt flag of interrupt flag register.
19. Then setting up of ADC is done, followed by selection of channel, Channel AIN4 is selected, and PB4 pin is used.
20. A potentiometer is connected so as to vary the voltage.
21. By the use of CR1 register of ADC, ADC is turned on and also continuous mode is set.

22. Right alignment is set by using CR2 register.
23. Then main() function is started, under main function firstly setup of ADC, output ports and timer is done followed by delays.
24. We stated PD4 as a floating input since potentiometer is to be connected for varying voltages.
25. While(1) loop is used since we need an infinite loop to be executed.
26. Under this loop “wait_for_interrupt()” function call is done, as soon as the interrupt arrives the program is redirected to the required cases defined.

Note- #prgma vector=8, this statement tells the compiler which interrupt vector we are going to be writing. In our case, it is vector 8 which is EXTI_PORTD interrupt vector.

References

RM0016 – STM8S105C6 user manual

STM8S Datasheet for connections.

OBSERVATION-

As STM8S105C6 has 10 bit Successive approximation ADC, the decimal equivalent varies from 0 to 1023. The ADC has 10 i/p channels

RESULT-

The LEDs are made to glow in the desired pattern with voltage varying using Potentiometer and ADC Interfacing.