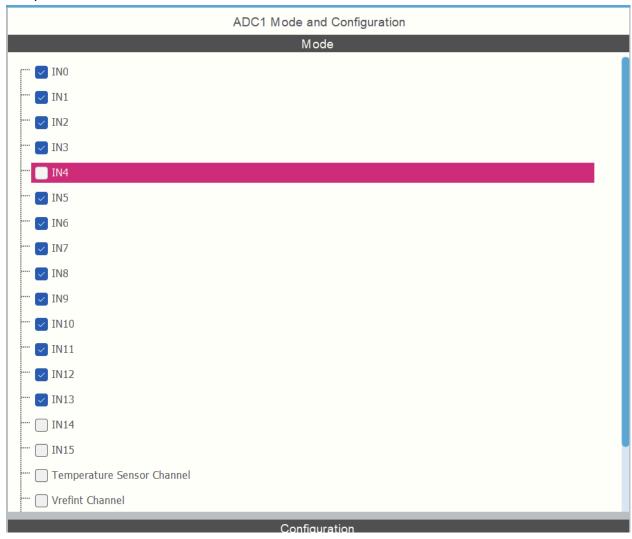
Setup the ADC as such

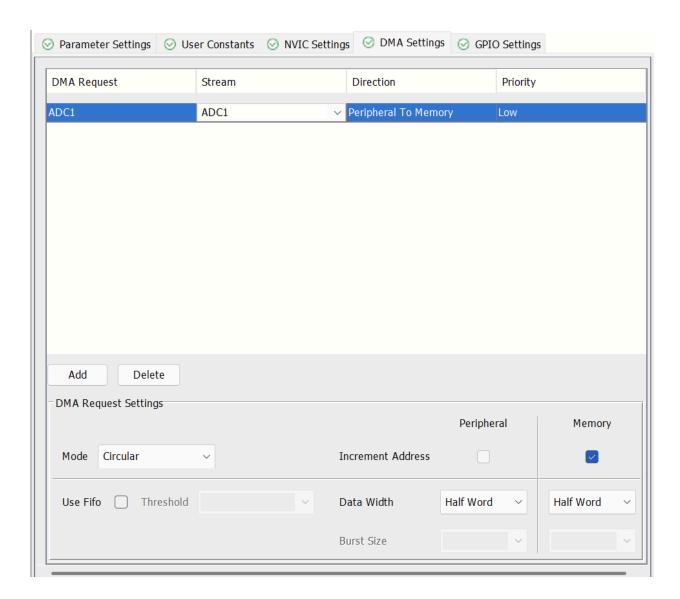


Mode	Independent mode	
ADC_Settings	<u> </u>	
Clock Prescaler	PCLK2 divided by 4	
Resolution	12 bits (15 ADC Clock cycles)	
Data Alignment	Right alignment Enabled Enabled Disabled	
Scan Conversion Mode		
Continuous Conversion Mode		
Discontinuous Conversion Mode		
DMA Continuous Requests	Enabled	
End Of Conversion Selection	EOC flag at the end of all conversions	
ADC_Regular_ConversionMode		
Number Of Conversion	13	
External Trigger Conversion Source	Regular Conversion launched by software	
External Trigger Conversion Edge	None	
Rank	1	
Channel	Channel 0	
Sampling Time	56 Cycles	
Rank	2	
Channel	Channel 1	
Sampling Time	56 Cycles	
Rank	3	
Channel	Channel 2	

_	Rank	3
	Channel	Channel 2
	Sampling Time	56 Cycles
	Rank	4
	Channel	Channel 3
	Sampling Time	56 Cycles
	Rank	5
	Channel	Channel 5
	Sampling Time	56 Cycles
_	Rank	6
	Channel	Channel 6
	Sampling Time	56 Cycles
_	Rank	7
	Channel	Channel 7
	Sampling Time	56 Cycles
	Rank	8
	Channel	Channel 8
	Sampling Time	56 Cycles
	Rank	9
	Channel	Channel 9
	Sampling Time	56 Cycles
	Rank	10
	Channel	Channel 10

	Sampling Time	56 Cycles
	Rank	8
	Channel	Channel 8
	Sampling Time	56 Cycles
	Rank	9
	Channel	Channel 9
	Sampling Time	56 Cycles
	Rank	10
	Channel	Channel 10
	Sampling Time	56 Cycles
	Rank	11
	Channel	Channel 11
	Sampling Time	56 Cycles
	Rank	12
	Channel	Channel 12
	Sampling Time	56 Cycles
	Rank	13
	Channel	Channel 13
	Sampling Time	56 Cycles
■ ADC_Inject	ted_ConversionMode	
Num	nber Of Conversions	0
─ WatchDog		
Ena	ble Analog WatchDog Mode	

NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priority
ADC1, ADC2 and ADC3 global interrupts	$\overline{\mathbf{v}}$	0	0
DMA2 stream0 global interrupt		0	0



Examples

Initialization

```
struct servoAngleReader servo_angle_reader;
servo_angle_reader.servoADC = &hadc1;
servoReaderStart(&servo_angle_reader);
```

Create a servoAngleReader variable and initialize the servoADC to the ADC used to read the servo angles.

Start the ADC conversions

Reading Angles

```
updateAngles(&servo_angle_reader);
servo_angle_IN10 = getAngleValue(&servo_angle_reader, NECK_CH);
```

Update the angles first by calling updateAngles. Then, use getAngleValue to read a certain servo's angle. The return type is double

```
#define LB_TIBIA_CH 1
#define LB_FEMUR_CH 0
#define LB_SHOUL_CH 7
#define LF_TIBIA_CH 11
#define LF_FEMUR_CH 10
#define LF_SHOUL_CH 8
#define RF_TIBIA_CH 3
#define RF_FEMUR_CH 2
#define RF_SHOUL_CH 5
#define RB_TIBIA_CH 4
#define RB_TIBIA_CH 4
#define RB_FEMUR_CH 12
#define RB_SHOUL_CH 6
#define RB_SHOUL_CH 6
```

The macro in the .h file contains the necessary channel assignments