2.10 INTERNAL DAC OF LPC1768

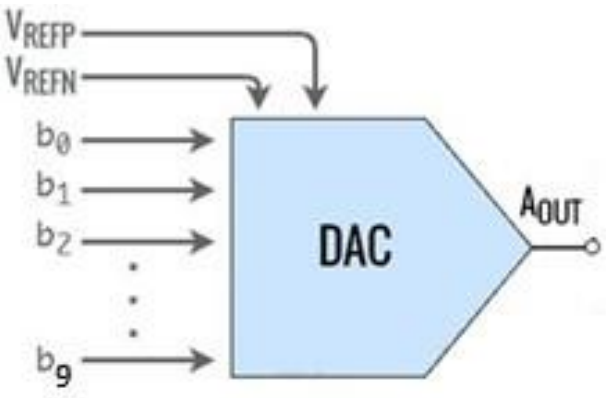
The LPC1768 microcontroller, based on the ARM Cortex-M3 core, features an Internal Digitalto-Analog Converter (DAC). This DAC allows you to convert digital values (binary data) to analog voltage signals, which is useful for audio output, waveform generation, or other analog applications.

2.10.1 FEATURES

- **Resolution:** 10-bit resolution (values from 0 to 1023).
- **Output Voltage Range:** From 0V to VREF (reference voltage).
- **Update Rate:** Up to 1 MHz.
- **Single-channel output:** One output channel only (through PIN AOUT - P0.26).
- **Selectable reference voltage:** Usually powered by VREF or internal supply voltage.

2.10.2 PIN DESCRIPTION

Pin	Type	Description
AOUT	Output	**Analog Output.** After the selected settling time after the DACR is written with a new value, the voltage on this pin (with respect to V_{SSA}) is VALUE × ((V_{REFP} - V_{REFN})/1024) + V_{REFN}.
V_{REFP}, V_{REFN}	Reference	**Voltage References.** These pins provide a voltage reference level for the ADC and DAC. **Note: V_{REFP} should be tied to VDD(3V3) and V_{REFN} should be tied to V_{SS} if the ADC and DAC are not used.**
V_{DDA}, V_{SSA}	Power	**Analog Power and Ground.** These should typically be the same voltages as V_{DD} and V_{SS}, but should be isolated to minimize noise and error. **Note: VDDA should be tied to VDD(3V3) and VSSA should be tied to VSS if the ADC and DAC are not used.**



- LPC 1768 has 10-bit internal DAC whose output is connected to P0.26
- V_{REFN} is connected to 0 V and V_{REFP} is connected to 3.3V.

2.10.3 PIN SELECT REGISTER 1 (PINSEL1)

PINSEL1	Pin name	Function when 00	Function when 01	Function when 10	Function when 11	Reset value
1:0	P0.16	GPIO Port 0.16	RXD1	SSEL0	SSEL	00
3:2	P0.17	GPIO Port 0.17	CTS1	MISO0	MISO	00
5:4	P0.18	GPIO Port 0.18	DCD1	MOSI0	MOSI	00
7:6	P0.19[1]	GPIO Port 0.19	DSR1	Reserved	SDA1	00
9:8	P0.20[1]	GPIO Port 0.20	DTR1	Reserved	SCL1	00
11:10	P0.21[1]	GPIO Port 0.21	RI1	Reserved	RD1	00
13:12	P0.22	GPIO Port 0.22	RTS1	Reserved	TD1	00
15:14	P0.23[1]	GPIO Port 0.23	AD0.0	I2SRX_CLK	CAP3.0	00
17:16	P0.24[1]	GPIO Port 0.24	AD0.1	I2SRX_WS	CAP3.1	00
19:18	P0.25	GPIO Port 0.25	AD0.2	I2SRX_SDA	TXD3	00
21:20	P0.26	GPIO Port 0.26	AD0.3	AOUT	RXD3	00
23:22	P0.27[1][2]	GPIO Port 0.27	SDA0	USB_SDA	Reserved	00
25:24	P0.28[1][2]	GPIO Port 0.28	SCL0	USB_SCL	Reserved	00

By default the pins are configured as GPIO pins. To select P0.26 for AOUT functionality write “10” to bits 21:20 of PINSEL1 register.

```
LPC_PINCON->PINSEL1 |= (1 << 21); //write 1
LPC_PINCON->PINSEL1 &= ~(1 << 20); //write 0
```

2.10.4 D/A CONVERTER REGISTER (DACR REGISTER)

This read/write register includes the digital value to be converted to analog, and a bit that trades off performance vs. power. Bits 5:0 are reserved for future, higher-resolution D/A converters.

Bits	Name	Description
[15:6]	VALUE	10-bit value (0-1023) for output.
[16]	BIAS	Set to 1 for lower power (default).
[31:17]	-	Reserved

DAC Output Voltage Formula:

$$V_{out} = \text{DACValue} \times \frac{V_{ref}}{2_{10}} - 1$$

Where:

DACValue: 10-bit value from 0 to 1023

V_{REF}: Reference voltage (typically 3.3V)

Example:

a) DAC_Value = 1023 $V_{out} = 3.3$ V

b) DAC_Value = 512 $V_{out} = 1.65$ V

- **DACR[16]** controls the **BIAS of DAC** ○ If DACR[16] = 0 then DAC output settles within 1 µs consuming a maximum current of 700 µA.
  ○ If DACR[16] = 1 the DAC output settles within 2.5 µs consuming a maximum current of 300 µA - Low Power (Default)

2.10.5 EXAMPLE CODE 01 – SQUARE WAVE GENERATION

Write a C program for the LPC17xx microcontroller that generates a square wave of desired amplitude using internal DAC.

```
#include <lpc17xx.h>
#define AMPLITUDE 3.3
 void delay(unsigned int
ms)
{
     unsigned int i,j;
     for(i=0; i<ms; i++)
     for(j=0; j<1275; j++);
}
int main(void)
{
     unsigned long int dacrValue = 0;
     unsigned long int dac_value = 0;

      SystemInit();

      //Configure P0.26 for DAC Analog Output
     LPC_PINCON->PINSEL1 |= (1 << 21);       LPC_PINCON-
>PINSEL1 &= ~(1 << 20);       //Select low power

      LPC_DAC->DACR |= (1 << 16);

     //DAC Value          dac_value =
AMPLITUDE * 1023 / 3.3;
```

```
	while(1)
	{
		//Active High Pulse
		dacrValue &= ~(0x3FF << 6);	//clear DAC bits
	dacrValue |= (dac_value << 6); //write from bit position 6
		LPC_DAC->DACR = dacrValue;	//write to DAC register
		delay(100);

		//Active Low Pulse		dacrValue &= ~(0x3FF <<
6);  //clear DAC bits		LPC_DAC->DACR = dacrValue;
//write to DAC register		delay(100);

	}
}
```

2.10.6 EXAMPLE CODE 02 – TRIANGULAR WAVE GENERATION

Write a C program for the LPC17xx microcontroller that generates a triangular waveform using internal DAC.

```
#include <lpc17xx.h>
 int
main(void)
{
	short int i = 0;
	unsigned long int dacrValue = 0;

	SystemInit();

	//Configure P0.26 as DAC output pin	LPC_PINCON-
>PINSEL1 |= (1 << 21);
	LPC_PINCON->PINSEL1 &= ~(1 << 20);

	// Low Power
	LPC_DAC->DACR |= (1 << 16);
```

```
    while(1)
    {
        for(i = 0; i < 1024; i++)
        {
            dacrValue &= ~(0x3FF << 6); //clear bits
    dacrValue |= (i << 6);   //write from bit position 6
            LPC_DAC->DACR = dacrValue;    //write to DAC register
        }

        for(i = 1023; i > 0; i--)
        {
            dacrValue &= ~(0x3FF << 6); //clear bits
    dacrValue |= (i << 6);   //write from bit position 6
            LPC_DAC->DACR = dacrValue;    //write to DAC register
        }
    }
}
```

2.10.7 EXAMPLE CODE 03 – SINE WAVE GENERATION

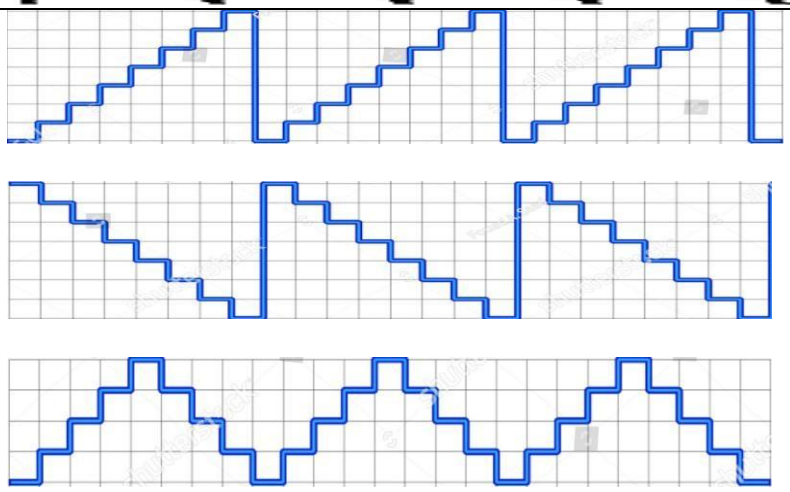
Write a C program for the LPC17xx microcontroller that generates a sine waveform using internal DAC.

```
#include <lpc17xx.h>
#include <math.h>

#define PI          3.14159265
#define SAMPLES     1024
 int
main(void)
{    short int i = 0;    unsigned
long int dacrValue = 0;       float
angle = 0;

    SystemInit();

    //Configure P0.26 as DAC output pin
    LPC_PINCON->PINSEL1 |= (1 << 21);
    LPC_PINCON->PINSEL1 &= ~(1 << 20);
```

```
      //low power configuration
      LPC_DAC->DACR |= (1 << 16);

      while(1)
      {
           for(i = 0; i < SAMPLES; i++)
           {
             // Generate sine wave value from 0 to 1023
       angle = (2 * PI * i) / SAMPLES;

             // Scale to 0-1023 range
             dacrValue = (unsigned long int)((sin(angle) + 1) * 511.5);

             // Write to DAC register
             LPC_DAC->DACR = (dacrValue << 6);
           }
      }
}
```

Further exploration:

1) Generate the following waveforms using internal DAC

Rising Sawtooth	
Falling Sawtooth	
Staircase waveforms	

2) Do you know how to use timers of LPC1768?. If yes, generate waveforms for specific frequency using timers.