8. Digital to analog converters

There are two 12-bit digital to analog converters (DAC) available. A principal block diagram for one of the two identical DACs is given in Fig. 1 (a copy of Fig. 48, RM0090, page 251).

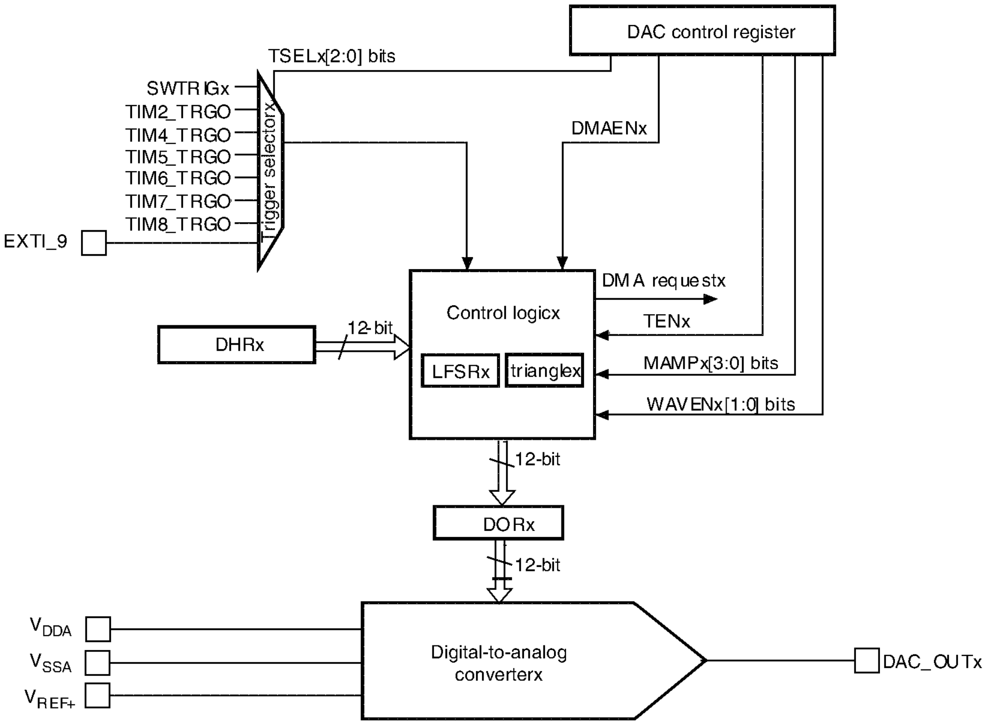

Figure 1: The block diagram of a single DAC

The actual DAC is shown at the bottom. Its operation is supported by power supply lines and the reference signal, all coming from pins of the microcontroller on the left. The output of the DAC is available at a pin to the right. The user program loads data (to be converted to analog signal) for the DAC into the data holding register DHR from where the data is transferred into register DOR to be instantly converted into analog signal by the DAC.

The register DHR is a 32-bit wide register, but can be accessed as a whole (DAC_DHR12RD, 12 bits for each DAC, data right aligned in both halves of the 32-bit register) or each half separately (DAC_DHR12R1 and DAC_DHR12R2, 12 bits for each DAC1 or DAC2, data right aligned in a 32-bit register). Each half of the register DHR is responsible for one DAC.

The moment of transferring the data from register DHR into DOR can be automatic, but can also be defined by software (SWTRIGx), hardware (TIMx_TRGO) or external signals (EXTI_9), as selected by the multiplexor on the left above the register DHR and defined by bits TSELx[2:0]. Such organization allows the synchronization of signals out of the two DAC with a selected event. There is also a possibility to transfer the data using a direct memory access (DMA, to be demonstrated later).

Additional hardware is provided in Control logicx box to generate noise or triangle wave signal to be added to the content of the register DHR.

Several bits distributed among registers DAC_CR (DAC Control Register), DAC_SWTRIGR (DAC software trigger register) and DAC_SR (DAC Status Register) define the operation of the two DACs. Individual bits are explained in RM0090, chapter 12.

The demo program has been prepared, and its listing is given in Fig. 2. The program uses two DACs to generate two different saw-tooth signals. It starts with the initialization of both DACs and continues with the infinite loop. Within the loop the data for two saw-tooth signals is calculated and written into registers DAC_DHR12R1 and DAC_DHR12R2

```
#include "stm32f4xx.h"

  /*

  outputs are at PA4 & PA5 -> pins must be enabled for analog!
                              GPIOA->MODER   |= 0x00000f00;
  */

int main ()  {
unsigned int j;

  RCC->APB1ENR |= 0x20000000;            // Enable clock for DAC

  DAC->CR |= 0x00010001;                 // DAC control reg, both channels ON
  GPIOA->MODER   |= 0x00000f00;          // MODE PortA, PA4 & PA5 are analog!

  while (1) {                            // infinite loop
    DAC->DHR12R1 = j & 0xfff;            // up ramp
    DAC->DHR12R2 = 0xfff - (j & 0xfff);  // down ramp
    j = (j + 1) & 0x0fff;                // calculate new value
  };

};
```

Figure 2: A listing of the program to utilize both DACs, the outputs are two saw tooth signals