Lab-STICC
CNRS UMR-6285

Contact: Johann Laurent

johann.laurent@univ-ubs.fr

http://www.lab-sticc.eu

FIR FILTER CMSIS

- **Use of the Library**
  - **1st step**: Add the path of include files for the linker and the compiler (project properties)
  - **2nd step**: Add the library name and also the path of the source codes. Be careful add a **« : »** in front of the library name (not for the path).
  - **3rd step**: Add the ARM_MATH_CMx symbol (x: corresponds to your architecture for example 7 for Cortex M7)
  - **4th step**: Add the #include <arm_math.h> in your main.c

FIR FILTER CMSIS

- Use of the DSP_Lib
  - *1ère step: Define a filter instance*
    - arm_fir_instance_f32 FIR_F32_Struct;

```
typedef struct
  {
uint16_t numTaps; // Number of taps in the filter (coeff)

float32_t *pState; // pointer to State structure of size (Nb_coeff + taille bloc -1)

float32_t *pCoeffs;  // pointer to the coeff structure of size Nb_Coeff

  } arm_fir_instance_f32;
```

FIR FILTER CMSIS

- *2nd step: Define a filter instance*
  - float32_t firStateF32[BLOCK_SIZE + NUM_TAPS - 1];
  - The size of the structure depends to the block size that we want to compute at each function call, depends to the tap number of the filter -1. There is '-1' because the first coeff is b0 and not b1.

FIR FILTER CMSIS

- *3rd step: fir_init function call*
  - void arm_fir_init_f32(arm_fir_instance_f32 * S, uint16_t numTaps, float32_t * pCoeffs, float32_t * pState, uint32_t blockSize)
  - Arm_fir_instance_f32 *S : pointer to a 3 fields structure (Tap number, pointer to State, pointer to coeff
  - Uint16_t numTaps: Tap number of the filter
  - Float32_t *pCoeff: pointer to coeff structure
  - Float32_t *pState: pointer to State structure
  - Uint32_t blockSize: number of samples computes per call

FIR FILTER CMSIS

```
/* Assign the numTaps value to the field num_Tab of the State structure*/
S->numTaps = numTaps;

/* Assign the pointer pCoeffs to the fiels pCoeffs  of the State structure*/
S->pCoeffs = pCoeffs;

/* Reset the state buffer whose the size is (blockSize + tap number of the filter -1)*/
memset(pState, 0, (numTaps + (blockSize - 1u)) * sizeof(float32_t));

/* Assign the pointer state to the filed pState of the State structure */
S->pState = pState;
```

FIR FILTER CMSIS

- *4th step: Call to arm_fir_f32 function*

- void arm_fir_f32(const arm_fir_instance_f32 * S,float32_t * pSrc,float32_t * pDst,uint32_t blockSize)

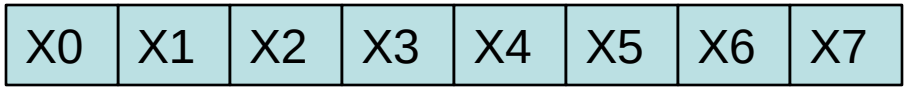
```
float32_t *pState = S->pState;          /* Assign the pointer pState to the pState field of the structure S */
float32_t *pCoeffs = S->pCoeffs;        /* Assign the pointer pCoeffs to the field pCoeff of the structure S*/
float32_t *pStateCurnt;                 /* Pointer to the current instance of the structure State */
float32_t *px, *pb;                     /* Temporary pointers on the State and to the Coeffs structures*/
float32_t acc0, acc1, acc2, acc3, acc4, acc5, acc6, acc7;     /* Accumulators */
float32_t x0, x1, x2, x3, x4, x5, x6, x7, c0;  /* Temporary variables to store the values of State and of coefficients*/
uint32_t numTaps = S->numTaps;          /* Filter tap number */
uint32_t i, tapCnt, blkCnt;             /* Loops counters*/

/* S->pState points to state array which contains previous frame (numTaps - 1) samples */
   /* pStateCurnt points to the location where the new input data should be written */
pStateCurnt = &(S->pState[(numTaps - 1u)]);
```

FIR FILTER CMSIS

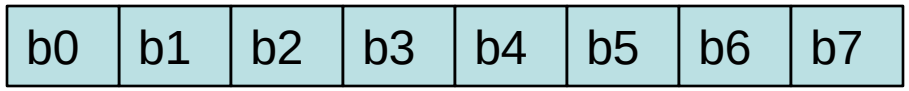
Algorithme de la fonction arm_fir_f32

1. The algo checks how many samples has to be computed per function call
    1. *By default it shares into n blocks of 8 output samples because each loop of the code (code unrolling) computes 8 outputs in parallel.*
2. The algo checks how many filter taps must be computed per iteration
    1. *By default it shares the filter coefficients into m blocks of 8 coefficients because each loop iteration (loop unrolling) uses 8 coefficients in parallel*
3. If the filter tap is not a multiple of 8 then the algorithm computes the number of taps it has left to process
4. Advance the state pointer by 8 to process the next group of 8 samples
5. The results in the 8 accumulators, store in the destination buffer
6. If the filter length is not a multiple of 8, compute the remaining filter taps
7. If the blockSize is not a multiple of 8, compute any remaining output samples
8. Now copy the last numTaps - 1 samples to the start of the state buffer.This prepares the state buffer for the next function call.

FIR FILTER CMSIS



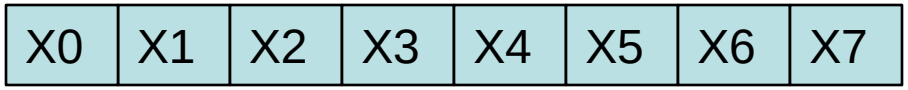
Samples



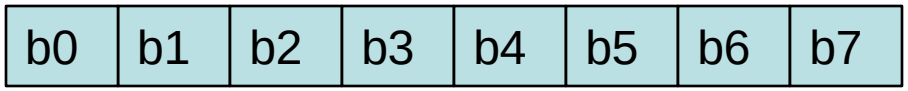
Coefficients

Acc0 = X0 b7
Acc1 = X1 b7
Acc2 = X2 b7
…….
Acc7 = X7 b7

FIR FILTER CMSIS



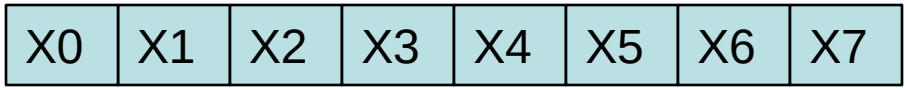
Samples



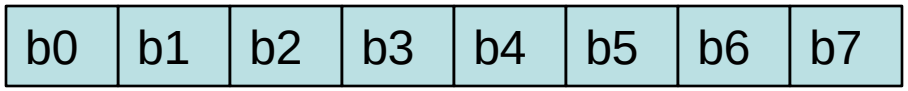
Coefficients

Acc0 = X0 b7 + X1 b6
Acc1 = X1 b7 + X2 b6
Acc2 = X2 b7 + X3 b6
…….
Acc7 = X7 b7 + X0 b6

FIR FILTER CMSIS



Samples



Coefficients

Acc0 = X0 b7 + X1 b6 + X2 b5 + X3 b4 + X4 b3 + X5 b2 + X6 b1 + X7 b0
Acc1 = X1 b7 + X2 b6 + X3 b5 + X4 b4 + X5 b3 + X6 b2 + X7 b1 + X0 b0
Acc2 = X2 b7 + X3 b6 + X4 b5 + X5 b4 + X6 b3 + X7 b2 + X0 b1 + X1 b0
…….
Acc7 = X7 b7 + X0 b6 + X1 b5 + X2 b4 + X3 b3 + X4 b2 + X5 b1 + X6 b0