

```

char cNewFrame = 0;
unsigned int iSampleCounter = 0;
fract16 inBuffer1[FRAMELENGTH], inBuffer2[FRAMELENGTH], outBuffer1[FRAMELENGTH], outBuffer2[FRAMELENGTH];
fract16 *pInFrame, *pOutFrame, *pInBuffer, *pOutBuffer; //Short

```

```

EX_INTERRUPT_HANDLER(Sport0_RX_ISR){
/* Add your own local variables here */

```

```

    // confirm interrupt handling
    *pDMA2_0_IRQ_STATUS = 0x0001;

```

```

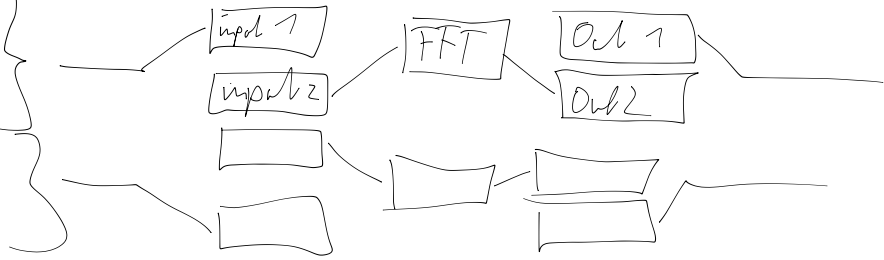
/* Add input and output and pointer setting here */

```

```

if(iSampleCounter == 0){
    if (pInFrame == inBuffer1){
        pInBuffer = inBuffer1;
        pInFrame = inBuffer2;
        pOutBuffer = outBuffer1;
        pOutFrame = outBuffer2;
    }else{
        pInBuffer = inBuffer2;
        pInFrame = inBuffer1;
        pOutBuffer = outBuffer2;
        pOutFrame = outBuffer1;
    }
    iDMATxBuffer[INTERNAL_DAC_L0] = POWER_OFF;
}
pInBuffer[iSampleCounter] = iDMARxBuffer[INTERNAL_ADC_R1]>>16; //Sample speichern
iDMATxBuffer[INTERNAL_DAC_R0] = pOutBuffer[iSampleCounter]<<16; //Speichern ausgeben
iSampleCounter++;
if (iSampleCounter>=FRAMELENGTH){
    iSampleCounter=0;
    iDMATxBuffer[INTERNAL_DAC_L0] = POWER_ON;
    cNewFrame = 1;
}
}

```



} setze zurück  
 //Sample speichern  
 //Speichern ausgeben

```

.section/DOUBLE32 program;

```

```

.align 2;
_winnmul:

```

```

    // void winmul(fract16 *pInFrame, fract16 *window, int ord);
    // R0 = Pointer to pInFrame (fract16)
    // R1 = Pointer to window (fract16)
    // R2 = FFT Order

```

```

    P1=R2; //Filterordnung
    I0=R0; //Signalwerte
    I1=R1; //Gewichtung; Fenster

```

```

    R3.L = 0; // Skalierungsfaktor

```

```

    NOP;NOP;NOP;NOP;

```

```

    R0.L=W[I0] || R1.L=W[I1++]; //Vorladen der ersten Signalwertes und Gewichtung

```

```

    LSETUP(_LOOP_START1,_LOOP_END1) LC0 = P1; //Schleife mit Filterordnung Durchlaufen
    _LOOP_START1:

```

```

        R2.L = R0.L * R1.L || R1.L = W[I1++]; //Fenster und Fensterkoeffizient annehmen
        R2.L = ASHIFT R2.L BY R3.L; //Skalieren
        W[I0++] = R2.L; //Abzeichen des letzten Signals
    _LOOP_END1:

```

```

        R0.L = W[I0]; //Anzeige wert

```

```

    RTS;

```

```

._winnmul.end:
.global _winnmul;

```