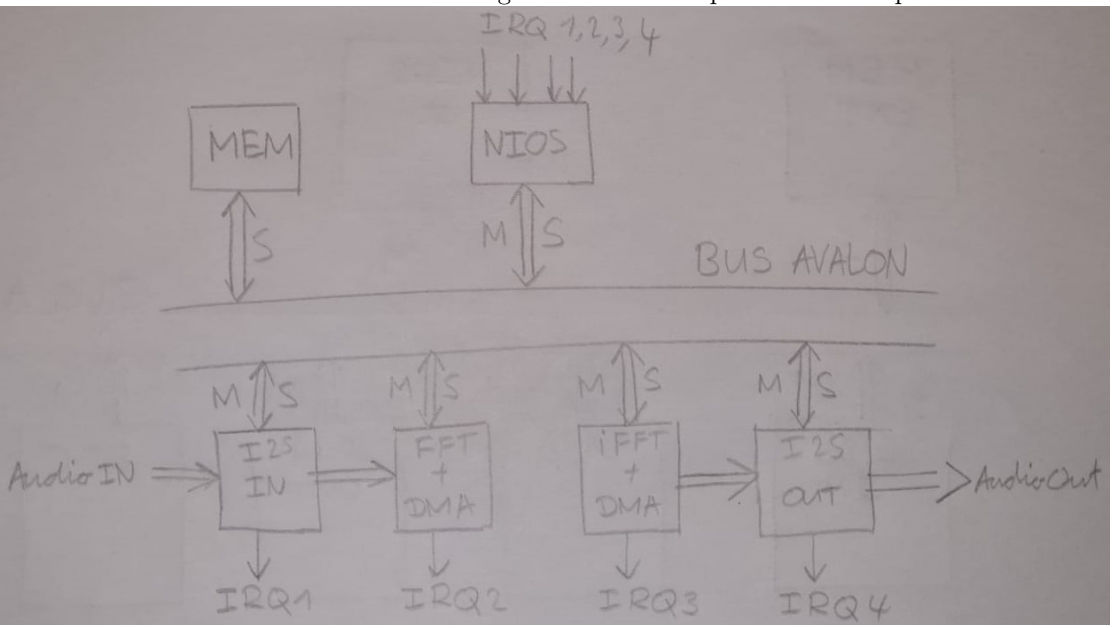
EmbHardw

Architecture

Exemple de shéma d'architecture:

1. Quand les données sont prêtes, I2S IN génère une IRQ
2. Le processeur répond à l'IRQ en activant le DMA pour transferer les données vers la mémoire.
3. Le processeur active le block FFT qui lis les données de la mémoire externe et génère une IRQ une fois la FFT terminée.
4. Le processeur effectue ses calculs à parir des données venant du block FFT.
5. Le processeur active le block iFFT qui les données du processeur et génère une IRQ une fois la iFFT terminée.
6. Le processeur répond à l'IRQ en activant le DMA pour transfert de données vers la mémoire.
7. I2S OUT génère une IRQ quand il est prêt à lire les données avec le DMA. Il lis les données et les envoie sur le bus I2S ensuite il génère une interrupttion une fois qu'il a fini.



Exemple d'identity port:

```
entity i2s_out is
    port (
        -- Avalon slave interface
        clock : in std_logic;
        reset : in std_logic;
        address : in std_logic_vector(1 downto 0);
        read : in std_logic;
        write : in std_logic;
        write_data : in std_logic_vector(31 downto 0);
        waitrequest : out std_logic;
        read_data : out std_logic_vector(31 downto 0);
        -- I2S interface for D/A
        sdim : out std_logic;
        lrclk : out std_logic;
        mclk : in std_logic;
        dem_sclk : in std_logic;
    );
end i2s_out;
```

Exemple de modèle de registre:

Reg Address	Reg name C	Reg name VHDL	R/W	Description
1	DATA	dmaData	W	En soft pour passer les données brutes, en hard comme data input
2	ADDR	dmaAddr	W	Addresse à atteindre par le DMA
3	SIZE	dmaSize	W	Taille du mot à transmettre
4	STATUS	dmaStatus	W	Dit si le DMA est libre
5	CONTROL	control	W	0 reset, 1 raw data, 2 channel when raw data or launch DMA, 3 anable interrupt
6	FREQ	freq	W	Fréquence d'opération de la communication sérielle

Différence streaming et memory mapped:
Le streaming permet de réduire la latence et la bandepassante nécessaire pour le transfert car les datas peuvent être envoyées et reçues en continu. La méthode memory-mapped est plus flexible et moins complexe à implémenter

Hardware Software co-design

Loop unrolling:
Loop unrolling consits of replicating the body of a loop to reduce the number of iterations.

```
for (i = 0; i < 3; i++) {
    for (j = 0; j < 3; j++) {
        a[i][j] = 0;
    }
}
```

$\Longleftrightarrow$

```
a[0][0] = 0;
a[0][1] = 0;
a[0][2] = 0;
a[1][0] = 0;
a[1][1] = 0;
a[1][2] = 0;
a[2][0] = 0;
a[2][1] = 0;
a[2][2] = 0;
```

In-lining:
In-lining consists of replacing a function call by the body of the function. The keyword `inline` can also be used to force the compiler to inline a function. The compiler option `-O3` enables in-lining.

```c
int sum(int a, int b) {
    return a + b;
}
int main() {
    int a = 1;
    int b = 2;
    int c = sum(a, b);
    return 0;
}
```

$\Longleftrightarrow$

```c
int main() {
    int a = 1;
    int b = 2;
    int c = a + b;
    return 0;
}
```

Using bit shift instead of multiplication/division:

Multiplication and division are very expensive operations in terms of clock cycles. It is therefore preferable to use bit shift when possible.

```c
int main() {
    int a = 1;
    int b = 2;
    int c = a * b;
    return 0;
}
```

$\Longleftrightarrow$

```c
int main() {
    int a = 1;
    int b = 2;
    int c = a << 1;
    return 0;
}
```

Computing inside the return statement:

```c
int main() {
    int a = 1;
    int b = 2;
    int c = a + b;
    return c;
}
```

$\Longleftrightarrow$

```c
int main() {
    int a = 1;
    int b = 2;
    return a + b;
}
```

Eviter les accès mémoire:

Par exemple rajouter des variables `*source` et `*dest` pour éviter les accès mémoire.

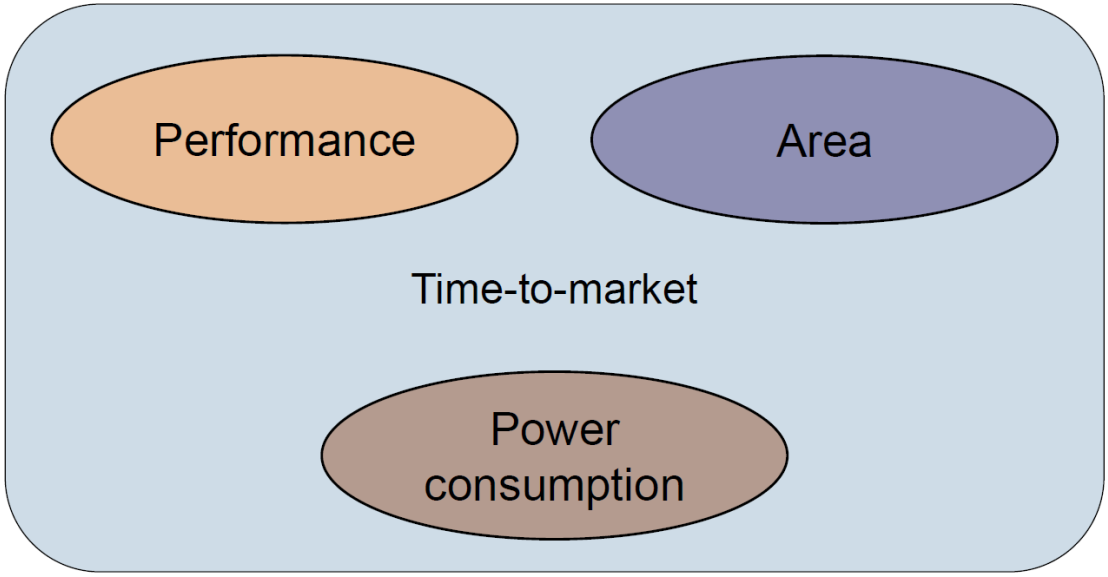
```c
int r,g,b;
int x,y,index;
for (y = 0; y < 480; y++) {
    for (x = 0; x < 640; x++) {
        index = y * 640 + x;
        r = source[index];
        g = source[index + 1];
        b = source[index + 2];
    }
}
```

$\Longleftrightarrow$

```c
int r,g,b;
int x,y,index;
int *source;
int *dest;
for (y = 0; y < 480; y++) {
    for (x = 0; x < 640; x++) {
        index = y * 640 + x;
        r = *source++;
        g = *source++;
        b = *source++;
    }
}
```

Trade-off

On peut remplacer certains bouts de code par du hardware en utilisant des custom instructions.



Facteurs limitants:

1. Compléxité de l'algorithme à remplacer
2. Dispoinibilité de ressources hardware
3. Temps de développement

Avantages:

1. Performance accrue
2. Consomation réduite

Cache:

Stocke les données les plus utilisées dans un espace mémoire plus rapide.

Scratchpad: Mémoire qui stock les données temporaires du processeur.

Problème de cohérence: Quand deux ou plusieurs processeurs essayent d'accéder à la même donnée en même temps. Résultats imprévisibles.

MSI (Modified, Shared, Invalid) est un protocole de cohérence de cache.

Mutex: permet de bloquer l'accès à une ressource partagée.

Profiling: Méthode pour analyser les performances d'un programme. Avec un timer on compte le nombre de cycles d'exécution d'une fonction. Il faut que le timer soit plus rapide que la fonction qu'on étudie et il faut un timer hardware.