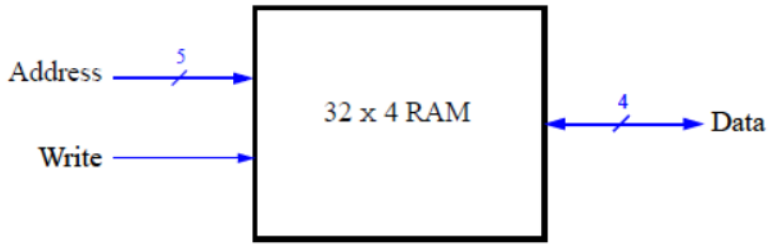
ENSA Marrakech **2022-2023**

TP1
Les Blocs Mémoires
GE41

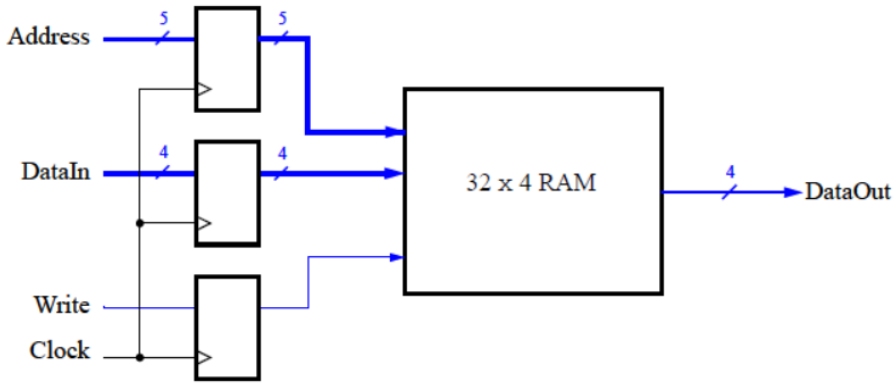
Dans les systèmes embarqués, il est nécessaire de prévoir une certaine quantité de mémoire. Si un système est mis en œuvre à l'aide de la technologie FPGA, il est possible de fournir des ressources de mémoire disponible dans la puce FPGA. Dans cet exercice, on examinera la mise en œuvre d'une telle mémoire.

Un schéma du module de mémoire vive (RAM) qu'on va implémenter est illustré à la figure 1a. Il contient 32 mots de quatre bits (lignes), auxquels on accède à l'aide d'un port d'adresse à cinq bits, d'un port de données à quatre bits et d'une entrée de commande d'écriture.

Le cyclone V, inclus sur la carte Intel DE1-SoC, contient des ressources mémoire dédiées qui sont des blocs M10K (on-chip). Chaque bloc M10K contient 10240 bits de mémoire. Les blocs M10k peuvent être configurés pour implémenter des mémoires de tailles différentes. La taille d'une mémoire est définie par sa profondeur en mots et sa largeur en bits (défini en anglais par le terme aspect ratio). Dans cet exercice, nous utiliserons une largeur de quatre bits et une profondeur de 32 mots.



a. Organisation de la RAM



b. Implémentation de la RAM

Fig 1: module de la RAM 32 x 4

Deux caractéristiques importantes des blocs M10K doivent être mentionnées. Premièrement, ils comprennent des registres qui peuvent être utilisés pour synchroniser tous les signaux d'entrée et de sortie sur une entrée d'horloge. Les registres sur les ports d'entrée doivent toujours être utilisés et les registres sur les ports de sortie sont facultatifs. Deuxièmement, les blocs ont des ports séparés pour les données écrites dans la mémoire et les données lues de la mémoire. Compte tenu de ces exigences, on va implémenter le module de RAM 32 x 4 modifié et qui est illustré à la figure 1b. Il comprend des registres pour les ports d'adresse, les entrée de données et l'entrée de contrôle de l'écriture, et utilise un port de sortie de données séparé et non registré (pas de registre à la sortie).

Partie 1

Des structures logiques couramment utilisées, telles que des additionneurs, des registres, des compteurs et des mémoires, peuvent être implémentées dans une puce FPGA en utilisant des modules prédéfinis fournis dans des librairies. Dans cet exercice, nous utiliserons un tel module pour implémenter la mémoire illustrée à la figure 1b.

1. Créer un nouveau projet Quartus pour implémenter le module de mémoire.

2. Cliquer sur **Tools > IP Catalog** afin d'ouvrir le catalogue des IPs dans le logiciel Quartus,. Dans la fenêtre **IP Catalog**, choisir le module **RAM: 1-PORT**, qui se trouve dans la catégorie **Basic Functions > On Chip Memory**. Sélectionner VHDL comme type de fichier de sortie à créer, nommer le fichier **OnChip_ram_32x4.vhd**, puis cliquer sur OK. Comme le montre la figure 2, spécifier une taille de mémoire de 32 mots de quatre bits. Sélectionner M10K. Également sur cette page ; accepter le paramètre par défaut pour utiliser une seule horloge pour les registres de la mémoire, puis passer à la page illustrée à la Figure 3. Sur cette page, désélectionner le paramètre appelé port de sortie « q » sous **Which ports should be registred?**. Ce paramètre crée un module RAM qui correspond à la structure de la figure 1b, avec des ports d'entrée registrés et des ports de sortie non registrés. Accepter les valeurs par défaut pour le reste des paramètres de l'assistant, puis cliquer sur le bouton **Finish** pour quitter cet outil. Examiner le fichier VHDL OnChip_ram_32x4.vhd qui définit le sous-circuit suivant :

```
ENTITY OnChip_ram_32x4.vhd EST
PORT (   address :      IN STD_LOGIC_VECTOR (4 DOWNTO 0);
         clock:         IN STD_LOGIC := '1';
         data :         IN STD_LOGIC_VECTOR (3 DOWNTO 0);
         wren :         IN STD_LOGIC ;
         q :            OUT STD_LOGIC_VECTOR (3 DOWNTO 0) );
END ram32x4 ;
```

3. Instancier ce sous-circuit dans un fichier VHDL de niveau supérieur qui inclut les signaux d'entrée et de la sortie pour les ports de mémoire indiqués dans la figure 1b.

4. Compiler le circuit. Observer dans le rapport de compilation que le compilateur Quartus utilise 128 bits dans l'un des blocs de mémoire FPGA pour implémenter le circuit RAM.

5. Simuler le comportement du circuit et s'assurer de pouvoir lire et écrire des données dans la mémoire. Un exemple de sortie de simulation est donné à la figure 4.

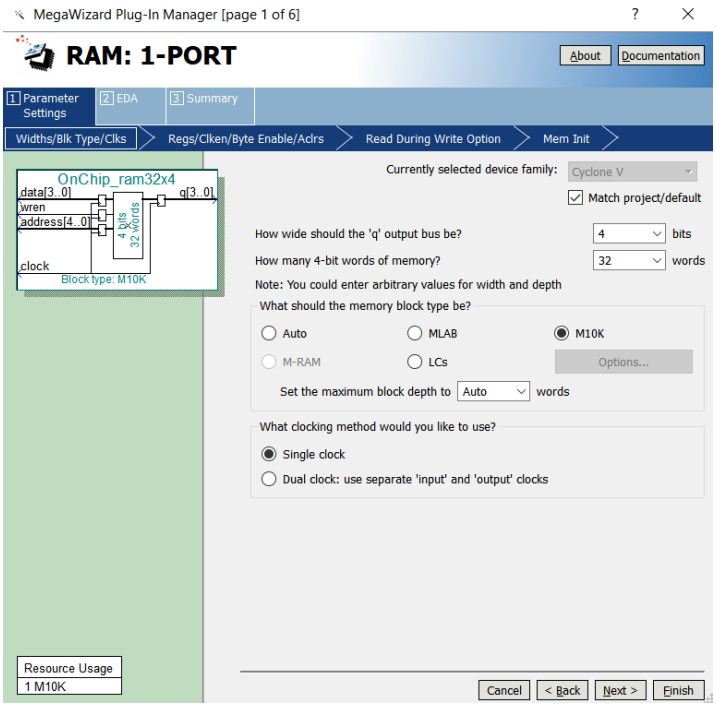


Fig. 2 : Configuration de la taille du module mémoire.

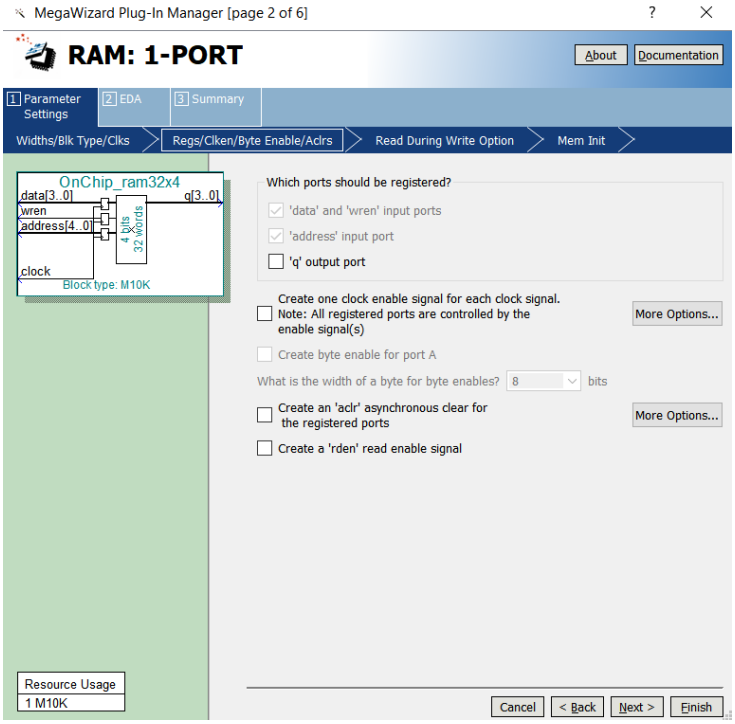


Fig.3 : Configuration des ports d'entrée et de sortie.

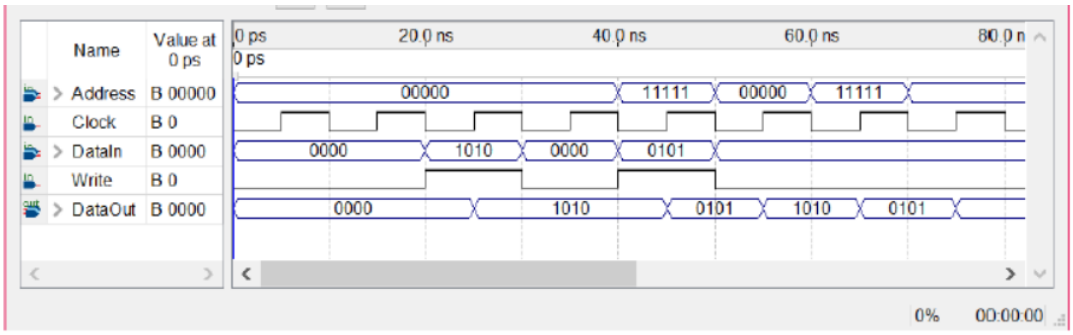


Fig.4: Exemple de simulation

Partie 2

Maintenant, on va réaliser le circuit de mémoire dans le FPGA et utiliser les switches de la carte DE1 SOC pour charger certaines données dans la mémoire créée. On souhaite également afficher le contenu de la RAM sur les afficheurs à 7 segments.

1. Créer un nouveau projet Quartus qui sera utilisé pour implémenter le circuit souhaité sur la carte.

2. Créer un autre fichier VHDL qui instancie le module ram32x4 et qui inclut les broches d'entrée et de sortie requises sur la carte. Utiliser les switches SW_{3-0} pour fournir des données d'entrée à la RAM et utiliser SW_{8-4} pour spécifier l'adresse. Utiliser SW_9 comme signal de contrôle de l'écriture et utiliser KEY_0 comme entrée d'horloge. Afficher la valeur d'adresse sur les afficheurs à 7 segments HEX5-4, afficher les données de l'entrée dans la mémoire sur HEX2 et afficher les données lues dans la mémoire sur HEX0.
En plus SW_{3-0}, SW_{8-4} et SW_9, doivent être affichées sur $LEDR_{3-0}$, sur $LEDR_{8-4}$ et sur $LEDR_9$ respectivement.

3. Tester le circuit et s'assurer que les données peuvent être stockées dans la mémoire à différentes adresses.

Partie 3

Au lieu de créer un sous-circuit de module de mémoire à l'aide du **IP CATALOG**, nous pouvons implémenter la mémoire requise en spécifiant sa structure en code VHDL. Il est possible de définir la mémoire comme un tableau bidimensionnel. Un tableau 32 x 4, qui a 32 mots avec 4 bits par mot, peut être déclaré par l'instruction

```
TYPE mem is ARRAY(0 TO 31) OF STD_LOGIC_VECTOR(3 DOWNTO 0);
SIGNAL memory_array : mem;
```

Dans un FPGA, un tel tableau peut être mis en œuvre soit en utilisant les bascules qui sont dans les éléments logiques, soit, plus efficacement, en utilisant les blocs de mémoire intégrés sur FPGA.
Effectuer les étapes suivantes :

1. Créer un nouveau projet qui sera utilisé pour implémenter le circuit souhaité sur la carte.

2. Compléter le code VHDL fourni dans le fichier **OnChip_ram_32x4_v2.vhd** pour réaliser les fonctionnalités nécessaires, comme cela a été fait dans la partie 2.

3. Attribuer les broches sur le FPGA pour connecter aux switches et aux afficheurs à 7 segments.

4. Compiler le circuit et le télécharger dans la puce FPGA.

5. Tester la fonctionnalité du design en appliquant certaines entrées et en observant la sortie.

Partie 4

Le bloc SRAM de la figure 1 possède un seul port qui fournit l'adresse pour les opérations de lecture et d'écriture. Pour cette partie, on va créer un type différent de module de mémoire, dans lequel il y a un port pour fournir l'adresse pour une opération de lecture, et un port séparé qui donne l'adresse pour une opération d'écriture. Effectuer les étapes suivantes.

1. Créer un nouveau projet Quartus pour le circuit. Pour générer le module mémoire souhaité, ouvrir **IP Catalog** et sélectionner le module **RAM : 2-PORT** dans la catégorie **Basic Functions > On Chip Memory**. Comme le montre la figure 5, choisir **With one read port and one write port** dans la catégorie intitulée **How will you be using the dual port ram?**

Configurer la taille de la mémoire, la méthode de synchronisation et les ports registrés de la même manière que dans la partie 2. Comme le montre la figure 6, sélectionner **I don't care (The outputs will be undefined)** pour **Mixed Port Read-During-Write for Single Input Clock RAM**. Ce paramètre spécifie qu'il importe peu que la mémoire a à sa sortie les nouvelles données en cours d'écriture ou les anciennes données précédemment stockées ; ceci correspond au cas des adresses d'écriture et de lecture sont les mêmes lors d'une opération d'écriture.

La figure 7 montre comment les mots mémoire peuvent être initialisés à des valeurs spécifiques. On utilise une fonctionnalité qui permet au module de mémoire d'être chargé avec des données lorsque le circuit est programmé dans la puce FPGA. Comme indiqué sur la figure, choisir le paramètre **Yes, use this file for the memory content data** et spécifier un nom de fichier (OnChip_ram_32x4.mif) . Un exemple de fichier MIF est fourni dans le dossier de travail sur le PC. Cliquer **Finish**, puis examiner le module de mémoire généré dans le fichier OnChip_ram_32x4.vhd.

2. Écrire un fichier VHDL qui instancie la mémoire à double port. Pour voir le contenu de la RAM, ajouter au design la capacité d'afficher le contenu de chaque mot de quatre bits (au format hexadécimal) sur l'afficheur à 7 segments HEX0. Utiliser un compteur comme adresse de lecture et faites défiler les emplacements de mémoire en affichant chaque mot pendant environ une seconde. Au fur et à mesure que chaque mot est affiché, indiquer son adresse (au format hexadécimal) sur les afficheurs à 7 segments HEX3-2. Utiliser l'horloge 50 MHz, CLOCK_50, et utiliser KEY_0 comme entrée de Reset. Pour l'adresse d'écriture et les données correspondantes, utiliser les switches SW_{8-4} et SW_{3-0}. Afficher l'adresse d'écriture sur HEX5-4 et afficher les données d'écriture sur HEX1. S'assurer de synchroniser correctement les switches d'entrée sur le signal d'horloge de 50 MHz.

3. Tester le circuit et vérifier que le contenu initial de la mémoire correspond au fichier OnChip_ram_32x4.mif. S'assurer de pouvoir écrire indépendamment des données sur n'importe quelle adresse à l'aide des switches.

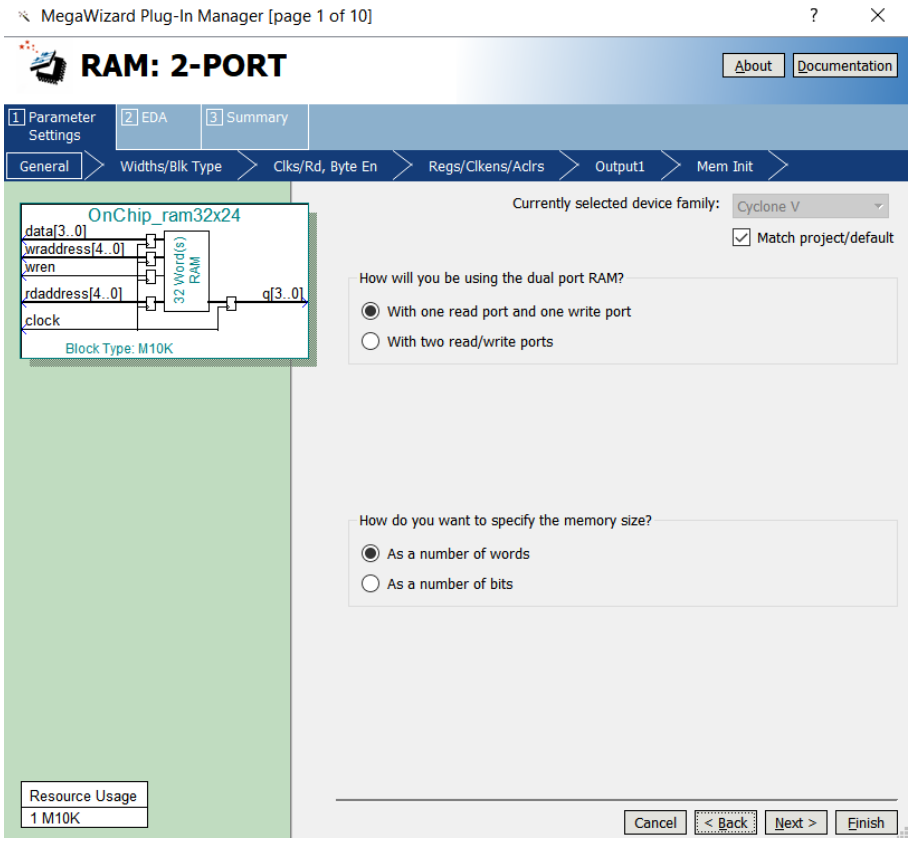


Fig 5 : Configuration des deux ports d'entrée de la RAM.

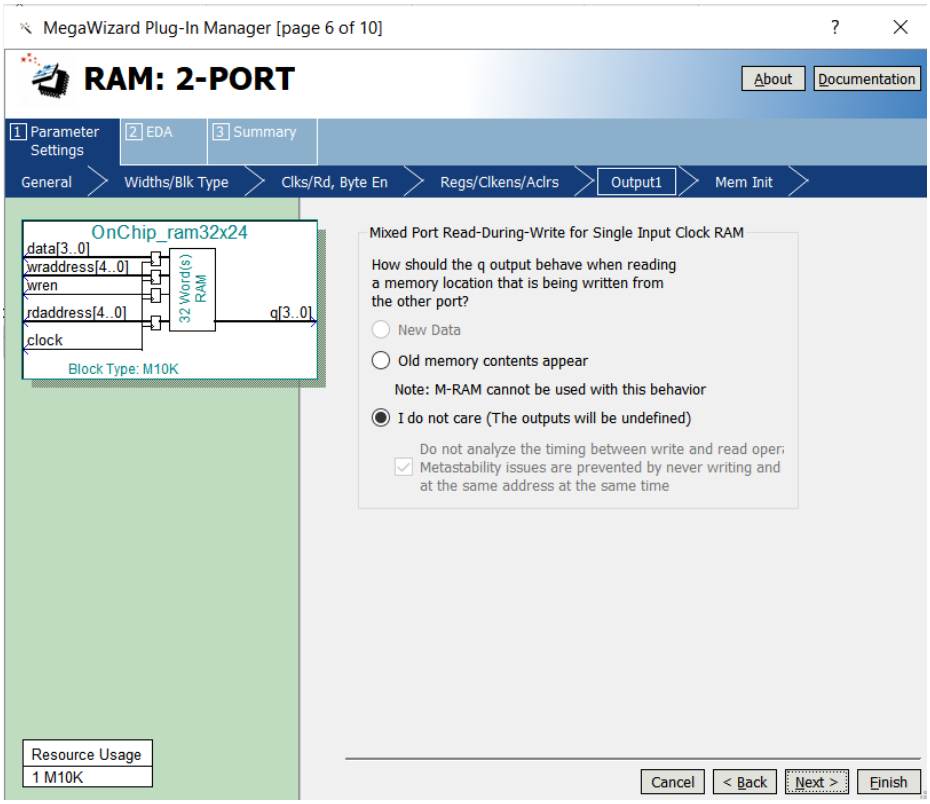


Fig. 6 : Configuration de la sortie de la RAM lors de la lecture et de l'écriture à la même adresse

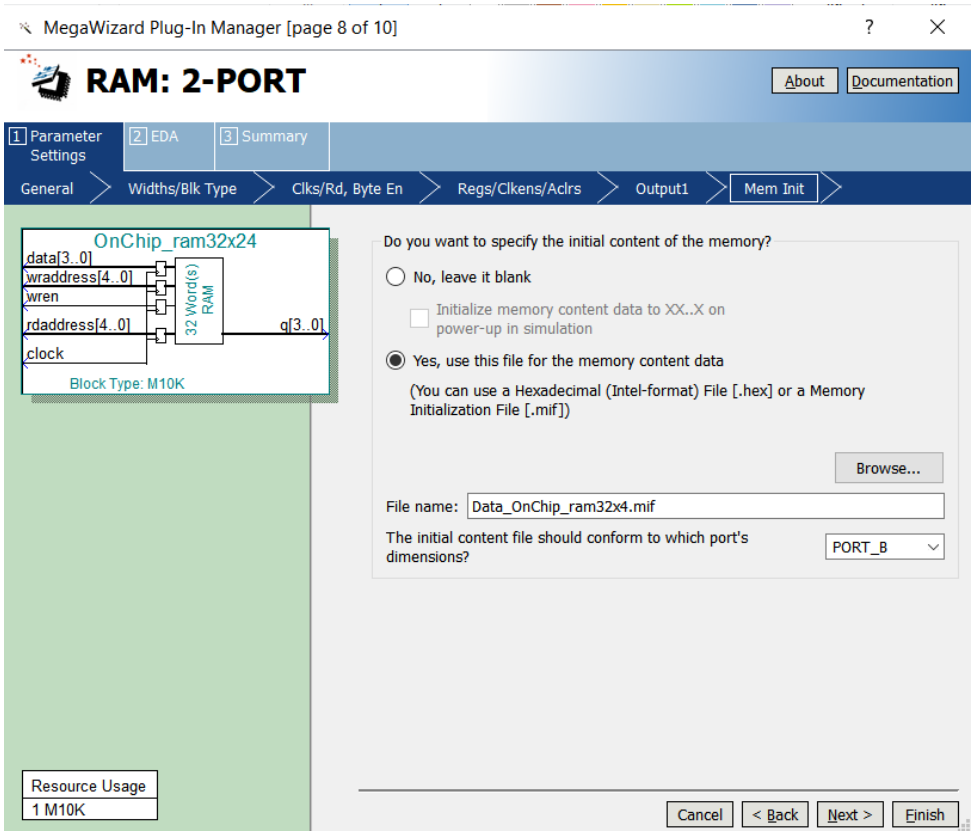


Fig. 7 : Spécification du fichier d'initialisation de la mémoire (MIF).