

Présentation du TP : L'objectif du TP est de mener à bien un projet d'implantation d'un réveil dans un circuit logique programmable. Il permettra de vous confronter aux diverses techniques de base de l'implantation d'application sur logique programmable via le langage VHDL. En outre, il permettra de vérifier votre maîtrise du logiciel Quartus II et de la mise en oeuvre de la logique programmable sur FPGA. On utilisera la carte DE2-115 de Terasic dotée d'un FPGA Altera Cyclone IV EP4CE115F29C7 .

Le réveil sera décrit en VHDL pour un affichage de l'heure, au format hhhmss soit 6 digits, et ce sur 0-24 heures. On ajoutera un afficheur pour indiquer par un A si le réveil est armé pour sonner ou non.

L'indication au logiciel du choix du positionnement des broches se fait par le biais d'un fichier script TCL. La syntaxe pour envoyer un signal sur une broche spécifique du FPGA est la suivante :

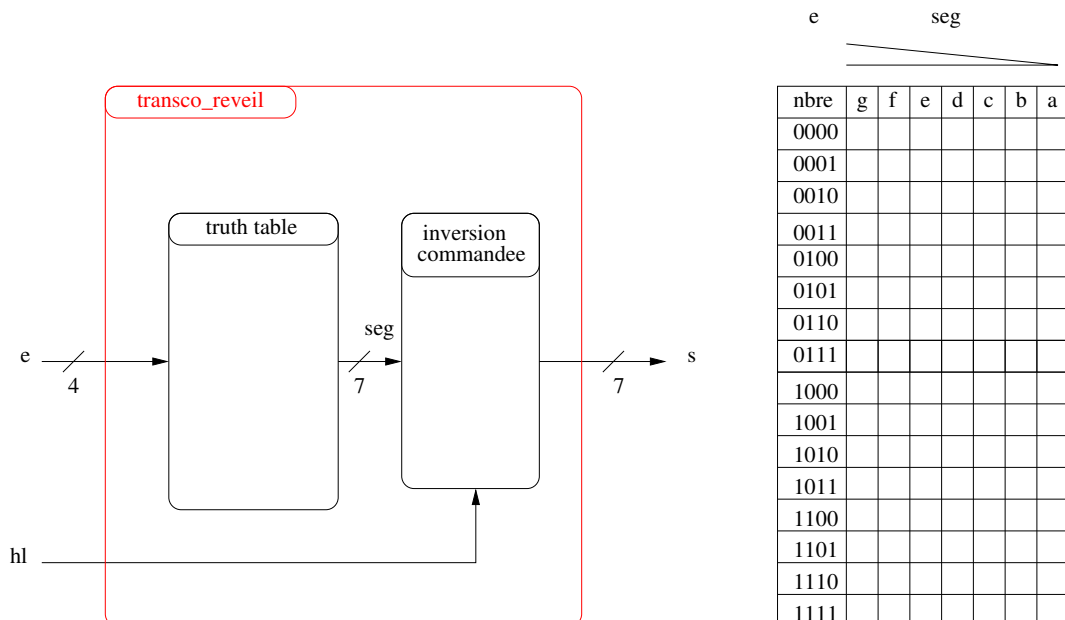
```
package require ::quartus::project
set_location_assignment PIN_Y2 -to clk
# ceci est un commentaire qui necessite un # au début de la ligne
# on demande donc d'envoyer le signal clk sur la broche Y2 du FPGA
```

Le script TCL devra être exécuté une fois au moins avant de compiler le projet. Il devra bien sûr être recompilé à chaque changement de son contenu, si l'on souhaite que ce changement soit pris en compte.

On vous donne en annexe un listing de script TCL sur lequel vous appuyer pour vos exercices. L'idée sera de mettre en commentaire certaines lignes et de laisser celles qui vous sont utiles valides.

Exercice 1 : transcodeur (logique combinatoire) On se propose de commencer par mettre en oeuvre le transcodeur qui va gérer les afficheurs 7 segments. On dessinera bien sûr les chiffres de 0 à 9 pour chaque nombre correspondant. On ajoutera le dessin du A pour la valeur 10 et on éteindra tous les segments pour les valeurs de 11 à 15.

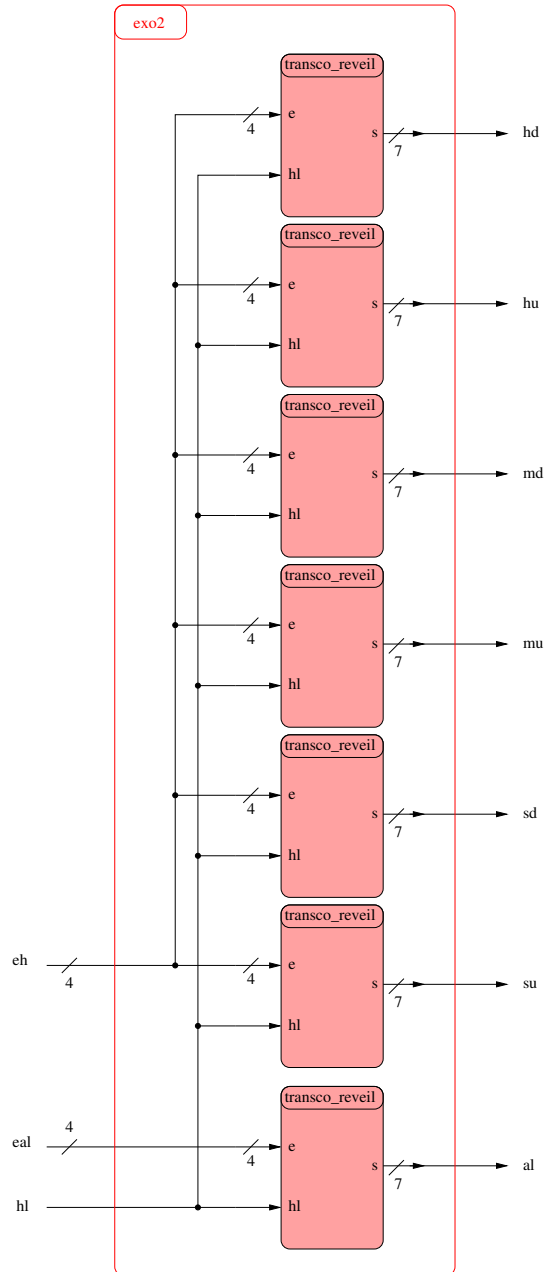
Préparation : remplir une table de vérité de ce transcodeur. Ne sachant pas, a priori, si on allume les segments par des 1 ou des 0 (anode ou cathode commune ?) , on choisit au hasard d'allumer les segments par des 1 . On ajoutera une deuxième fonction logique combinatoire à ce transcodeur pour inverser les sorties en fonction du positionnement d'une entrée supplémentaire nommée *h/l* . Le schéma synoptique est alors le suivant :



TP : Implanter le transcodeur en utilisant le style with select pour la table de vérité et un process pour l'inversion commandée dans le fichier *transco_reveil.vhd*. Préparer le fichier script *exo1.tcl* en utilisant l'afficheur hex0, les 4 switch sw[3:0] pour l'entrée et le switch sw[17] pour l'entrée *hl* . Vous vous appuyerez sur les indications de la présentation du TP et l'annexe pour générer ce fichier. Exécuter le script puis compiler le projet. Télécharger sur la carte et vérifier le bon fonctionnement. Faites valider par l'enseignant. Conserver les 2 fichiers sur une clé usb, dans votre bureau virtuel ou sur un serveur type "cloud" pour les prochains TP.

Exercice 2 : affichage_reveil On met ici en place l'affichage de l'heure complète, c'est à dire heures, minutes et secondes sur les afficheurs hex0 hex1 hex2 hex3 hex4 hex5. On ajoute ensuite l'affichage pour le A (de Alarme ou Armé) sur l'afficheur hex7. On choisit, pour tester l'application, d'envoyer la meme valeur d'entrée pour tous les afficheurs sauf pour celui du A. On organise la description VHDL sous la forme de **component** instanciés avec le style **port map**.

TP : Créer un nouveau projet et implanter cet affichage selon la figure ci-dessous en réutilisant les fichiers de l'exercice 1. Modifier le script .tcl en conséquence et l'exécuter avant de compiler le projet. Tester et faire valider par l'enseignant.

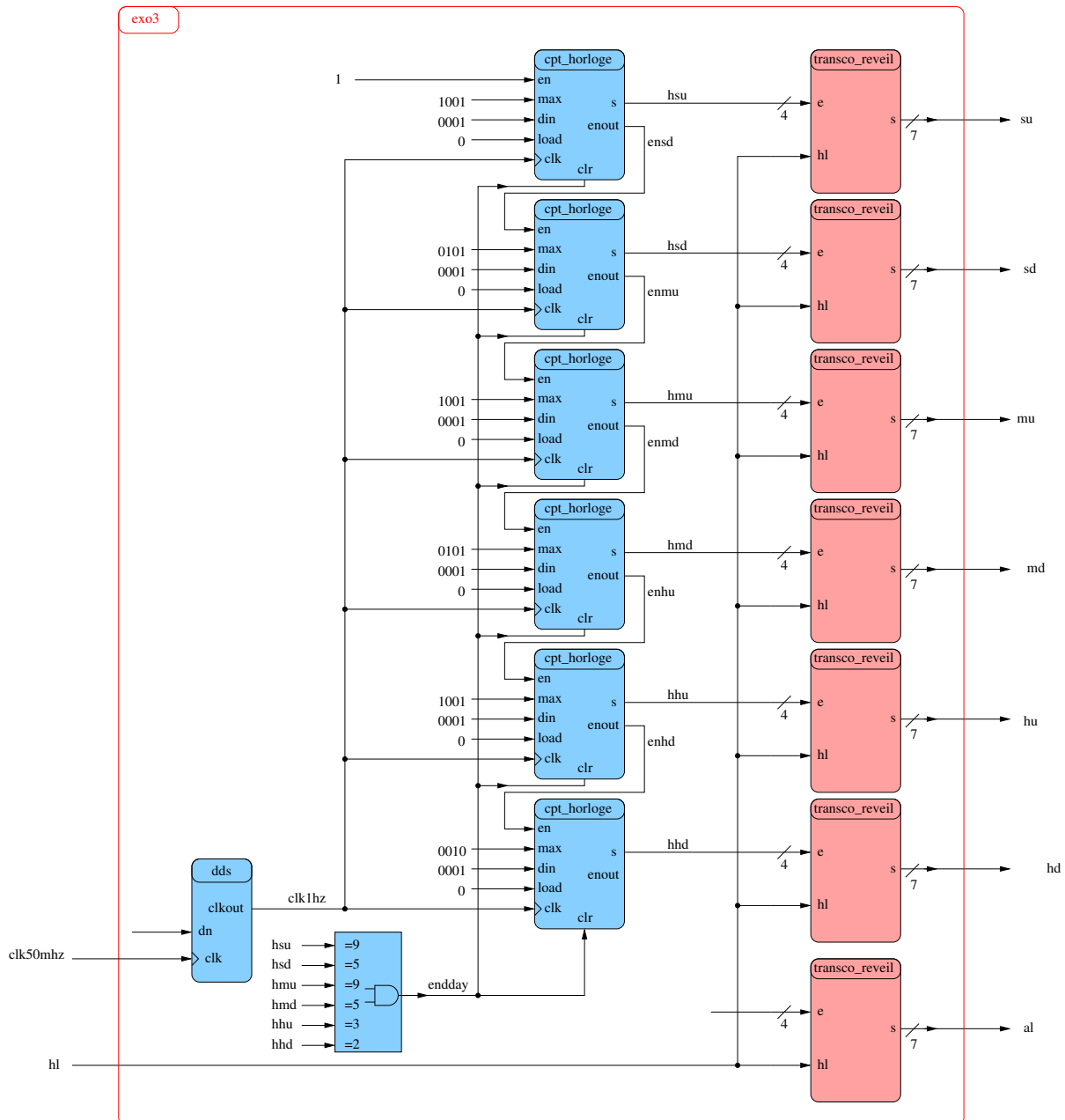


Exercice 3 : horloge On se propose maintenant de mettre en place les compteurs qui réalisent concrètement l'horloge. Ces compteurs doivent compter à la fréquence de 1Hz. En premier lieu, pour obtenir ce signal de référence à partir du quartz à 50Mhz de la carte, on met en place un composant nommé DDS (programme VHDL **dds.vhd** fourni) qui génère une fréquence proportionnelle à la fréquence d'entrée et à un coefficient d_n selon la formule suivante : $f_{out} = f_{in} \cdot \frac{d_n}{2^{32}}$. Les compteurs sont tous identiques à l'exception de la valeur maximale jusqu'à laquelle ils doivent compter. Ils sont placés en cascade pour compter les heures, les minutes et les secondes (pour chacun dizaines et unités, soit 6 compteurs)

TP :

- reprendre les programmes VHDL de l'exercice précédent et les insérer dans un nouveau projet.

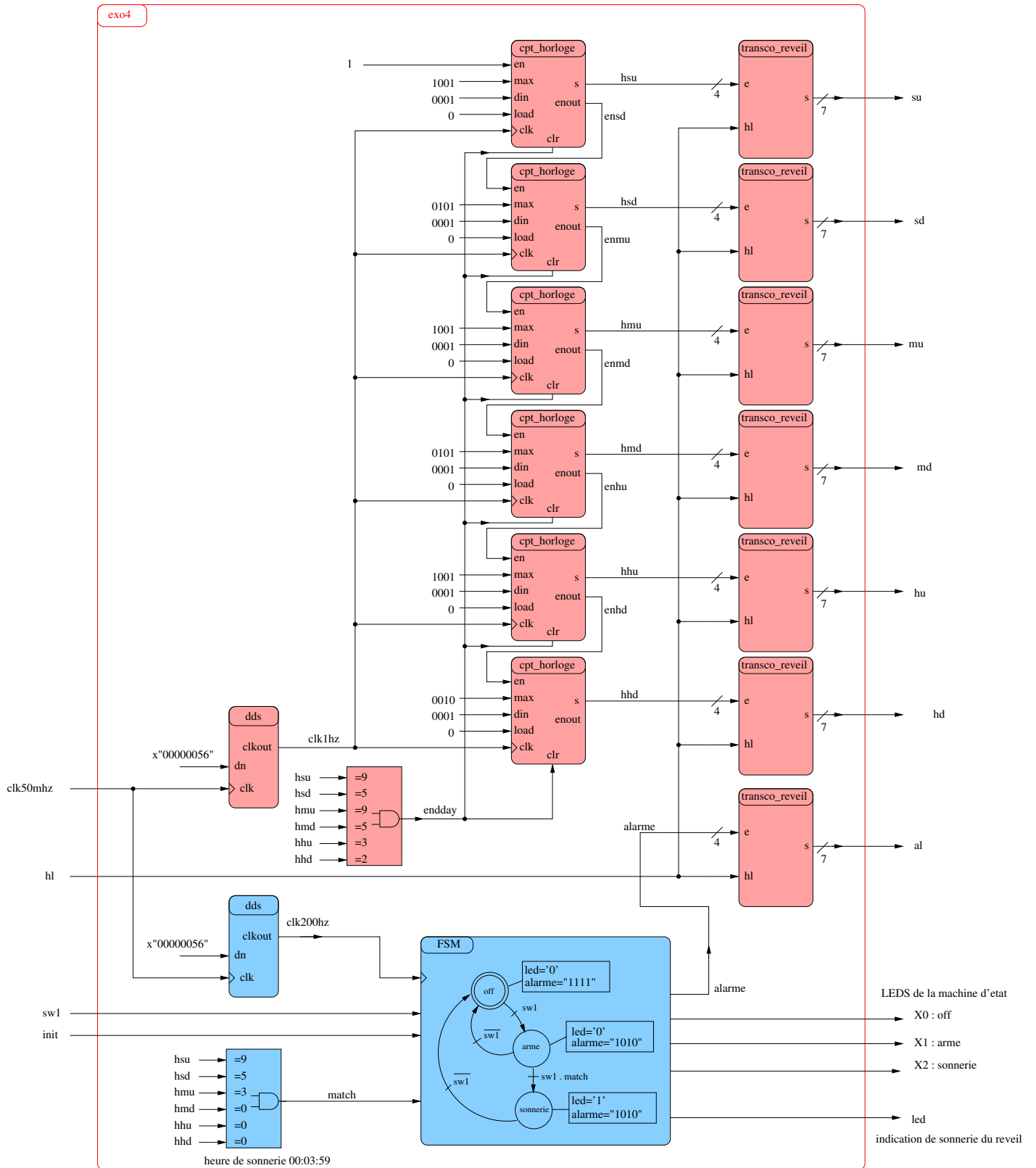
- ajouter le fichier ***dds.vhd*** à votre projet et ajouter la description de ce composant ***dds*** au programme VHDL “top-level” - voir le modèle pour ***transco-reveil*** . De même, ajouter ensuite la description du composant ***cpt_horloge***.
- Selon le schéma de la figure ci-dessous, implanter le ***dds*** dans l’architecture en fixant la valeur de ***dn*** pour obtenir non pas 1Hz mais 10Hz . Il s’agit de faire appel à l’instanciation de composant par ***port map*** .
- De même implanter dans l’architecture les 6 compteurs ***cpt_horloge*** . ***A ce stade de la programmation , vous pouvez faire l’implantation des compteurs, même si ces derniers ne sont pas encore décrits dans leur fonctionnement (architecture).***
- Terminer en décrivant un ***process*** combinatoire pour piloter la réinitialisation des compteurs à 0 quand on est en fin de journée (23:59:59). ***Faire contrôler le principe de ce programme VHDL par l’enseignant.***
- Décrire ensuite dans un fichier VHDL annexe (***cpt_horloge.vhd***) le comportement du compteur selon les contraintes suivantes :
 - la sortie ***enout*** est validée à 1 si l’entrée ***en*** vaut 1 avec la valeur du compteur qui est égale à sa valeur maximale autorisée.
 - la valeur du compteur , en sortie, évolue sur front montant d’horloge. Le comportement est le suivant :
 - * si le signal ***clr*** est actif (à 1) la valeur du compteur est forcée à zéro.
 - * sinon si l’entrée ***en=1*** alors si l’entrée ***load=1*** alors on charge le compteur avec la valeur ***din*** sinon si le compteur est inférieur à sa valeur maximale (***max***) alors on l’incrémente de 1 sinon on force le compteur à 0.
- implanter l’ensemble sur la carte DE2-115, tester et faire vérifier le bon fonctionnement par l’enseignant.



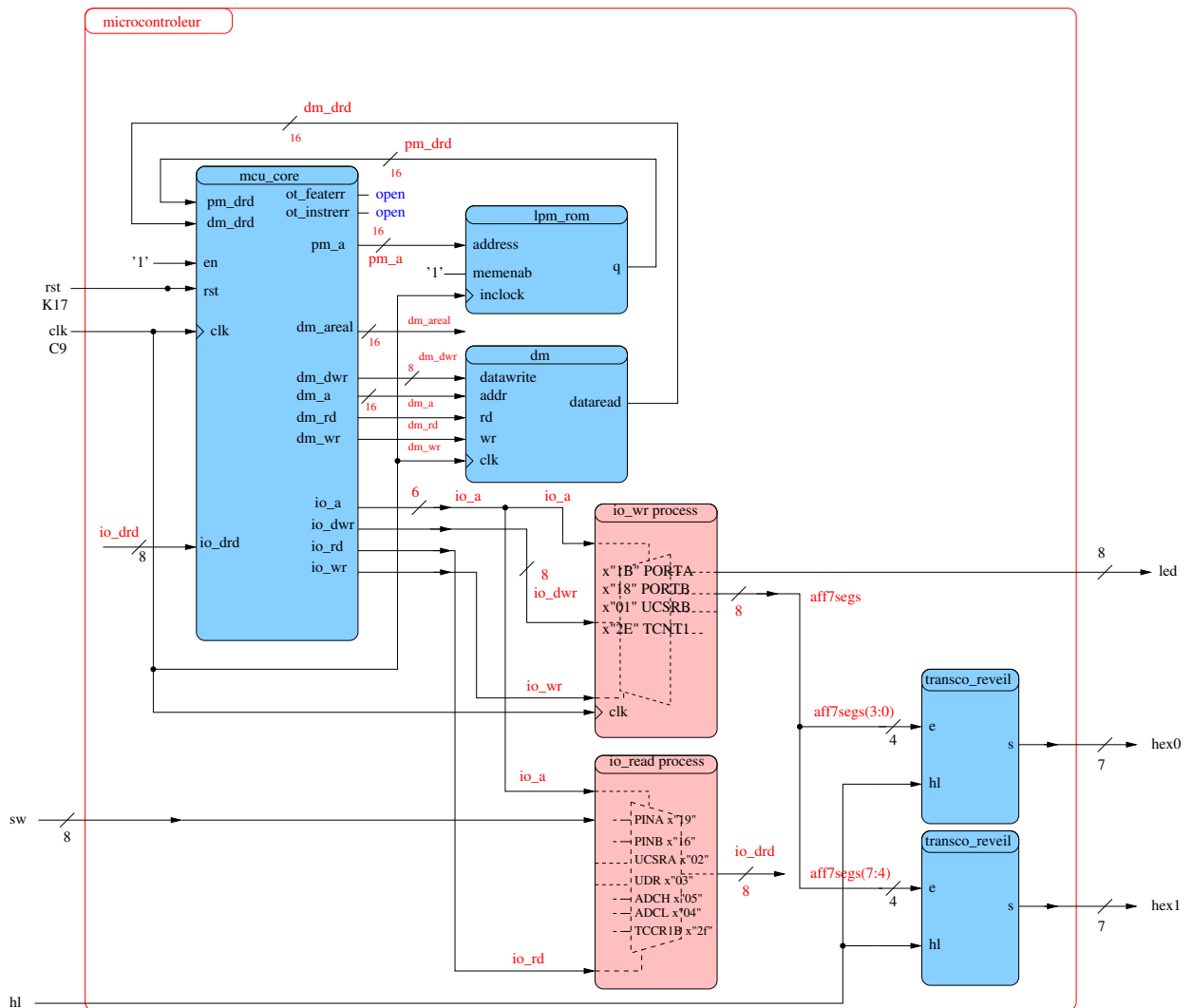
Exercice 4 : réveil On se propose de réaliser maintenant le réveil complet. On ajoute une machine d'état qui va gérer l'état du réveil ainsi que les fonctions logiques annexes pour faire retentir la sonnerie ou encore indiquer que le réveil est armé pour sonner. L'implantation complète du réveil et la description de la machine d'état sont représentées sur la figure ci-dessous.

TP : Reprendre, pour base de cet exercice, l'exercice 3 complet.

- Ajouter l'instanciation d'un générateur de fréquence (DDS) et le paramétrer pour générer une horloge à 200Hz pour piloter la machine d'état.
- Ajouter un process combinatoire sur le modèle de la détection de fin de journée (23:59:59) pour détecter l'instant de la sonnerie.
- Ajouter enfin la description de la machine d'état. On utilisera 3 leds pour visualiser à tout instant l'évolution de l'état de la machine d'état. On en utilisera une 4ième pour signifier la sonnerie du réveil à défaut de disposer d'un buzzer. On utilisera l'entrée sw1 pour armer et , désarmer le réveil, et aussi stopper la sonnerie.
- Compléter l'assignation des signaux d'entrée/sortie supplémentaire sur les broches du FPGA (outils Pin Planner et fichier .csv)
- Implanter et tester le fonctionnement du réveil. Faites valider par l'enseignant.



Exercice 5 : embarquer un processeur ATtiny L'objet de ce TP se "déconnecte" temporairement du projet du TP : le réveil. Il s'agit d'embarquer un premier processeur dans un FPGA et de bien s'appropriier tout la chaîne de développement que cela nécessite, ce qui reste un apprentissage non négligeable. La figure ci-dessous vous présente l'implantation à réaliser, sachant que vous disposez des ressources VHDL pour la description du processeur, de sa mémoire de données et du transcodeur. La mémoire programme est un composant disponible dans la librairie lpm. Quand aux process `io_read` et `io_write`, ils seront décrits directement en VHDL dans le programme VHDL principal (top).



TP :

- Implanter la partie matérielle selon ce schéma et suivez les instructions de l'enseignant pour les deux process. Exploiter le fichier .csv . Et compiler. Noter que la compilation est relativement longue...
- Copier les fichiers sources *test_tiny.c* , *comile_c.sh* et *mifwrite* depuis la ressource .zip sur le wikigeii vers le sous répertoire soft que vous aurez créé dans le répertoire de votre projet. Ouvrir et consulter ces fichiers .c et .sh .
- Compiler le source .c et recompiler ensuite tout le projet en notant que cette fois la compilation est plus rapide dès lors qu'on a sélectionné l'option **Use Smart Compilation** via le menu **Assignements->Settings->Compilation Process Settings**
- télécharger, vérifier le bon fonctionnement via le clignotement particulier des leds et l'affichage de **AA** et **55** sur les afficheurs 7 segments.

Annexe

Tableau de correspondance Ressources/FPGA Les principales ressources matérielles utilisées pour nos travaux sont : les horloges, les interrupteurs, les boutons poussoirs, les led, les afficheurs 7 segments, le connecteur de la liaison série, le connecteur VGA. Les noms des entrées/sorties, sont ceux de la documentation de la carte DE2-115 et correspondant à ce qui est indiqué en sérigraphie sur la carte. Dans les exercices, on utilise assez volontiers d'autres noms de signaux. On s'adaptera en conséquence...

| Signal Name | FPGA pin No | Description |
|-------------|-------------|-------------------|
| clock_50 | Y2 | 50Mhz clock input |
| clock2_50 | AG14 | 50Mhz clock input |
| clock3_50 | AG15 | 50Mhz clock input |
| sw[0] | AB28 | Slide Switch 0 |
| sw[1] | AC28 | Slide Switch 1 |
| sw[2] | AC27 | Slide Switch 2 |
| sw[3] | AD27 | Slide Switch 3 |
| sw[4] | AB27 | Slide Switch 4 |
| sw[5] | AC26 | Slide Switch 5 |
| sw[6] | AD26 | Slide Switch 6 |
| sw[7] | AB26 | Slide Switch 7 |
| sw[8] | AC25 | Slide Switch 8 |
| sw[9] | AB25 | Slide Switch 9 |
| sw[10] | AC24 | Slide Switch 10 |
| sw[11] | AB24 | Slide Switch 11 |
| sw[12] | AB23 | Slide Switch 12 |
| sw[13] | AA24 | Slide Switch 13 |
| sw[14] | AA23 | Slide Switch 14 |
| sw[15] | AA22 | Slide Switch 15 |
| sw[16] | Y24 | Slide Switch 16 |
| sw[17] | Y23 | Slide Switch 17 |
| key[0] | M23 | Push Button 0 |
| key[1] | M21 | Push Button 1 |
| key[2] | N21 | Push Button 2 |
| key[3] | R24 | Push Button 3 |
| ledr[0] | G19 | Led Red 0 |
| ledr[1] | F19 | Led Red 1 |
| ledr[2] | E19 | Led Red 2 |
| ledr[3] | F21 | Led Red 3 |
| ledr[4] | F18 | Led Red 4 |
| ledr[5] | E18 | Led Red 5 |
| ledr[6] | J19 | Led Red 6 |
| ledr[7] | H19 | Led Red 7 |
| ledr[8] | J17 | Led Red 8 |
| ledr[9] | G17 | Led Red 9 |
| ledr[10] | J15 | Led Red 10 |
| ledr[11] | H16 | Led Red 11 |
| ledr[12] | J16 | Led Red 12 |
| ledr[13] | H17 | Led Red 13 |
| ledr[14] | F15 | Led Red 14 |
| ledr[15] | G15 | Led Red 15 |
| ledr[16] | G16 | Led Red 16 |
| ledr[17] | H15 | Led Red 17 |
| ledg[0] | E21 | Led Green 0 |
| ledg[1] | E22 | Led Green 1 |
| ledg[2] | E25 | Led Green 2 |
| ledg[3] | E24 | Led Green 3 |
| ledg[4] | H21 | Led Green 4 |
| ledg[5] | G20 | Led Green 5 |
| ledg[6] | G22 | Led Green 6 |
| ledg[7] | G21 | Led Green 7 |
| ledg[8] | F17 | Led Green 8 |

| Signal Name | FPGA pin No | Description |
|-------------|-------------|------------------|
| vga_r[0] | E12 | vga red 0 |
| vga_r[1] | E11 | vga red 1 |
| vga_r[2] | D10 | vga red 2 |
| vga_r[3] | F12 | vga red 3 |
| vga_r[4] | G10 | vga red 4 |
| vga_r[5] | J12 | vga red 5 |
| vga_r[6] | H8 | vga red 6 |
| vga_r[7] | H10 | vga red 7 |
| vga_g[0] | G8 | vga green 0 |
| vga_g[1] | G11 | vga green 1 |
| vga_g[2] | F8 | vga green 2 |
| vga_g[3] | H12 | vga green 3 |
| vga_g[4] | C8 | vga green 4 |
| vga_g[5] | B8 | vga green 5 |
| vga_g[6] | F10 | vga green 6 |
| vga_g[7] | C9 | vga green 7 |
| vga_b[0] | B10 | vga blue 0 |
| vga_b[1] | A10 | vga blue 1 |
| vga_b[2] | C11 | vga blue 2 |
| vga_b[3] | B11 | vga blue 3 |
| vga_b[4] | A11 | vga blue 4 |
| vga_b[5] | C12 | vga blue 5 |
| vga_b[6] | D11 | vga blue 6 |
| vga_b[7] | D12 | vga blue 7 |
| vga_clk | A12 | vga clock |
| vga_blank_n | F11 | vga blank |
| vga_hs | G13 | horizontal sync |
| vga_vs | C13 | vertical sync |
| vga_sync_n | C10 | vga sync |
| I2C_sclk | B7 | I2C clock |
| I2C_sdat | A8 | I2C Data |
| UART_rxd | G12 | UART Receiver |
| UART_txd | G9 | UART Transmitter |
| UART_cts | G14 | Clear to send |
| UART_rts | J13 | Request to send |
| PS2 Clk | G6 | PS/2 Clock |
| PS2 dat | H5 | PS/2 Data |
| PS2 clk2 | G5 | 2nd device |
| PS2 dat2 | F5 | 2nd device |
| IRDA_rxd | Y15 | IR receiver |

| Signal Name | FPGA pin No | Description |
|-------------|-------------|-----------------|
| Hex0[0] | G18 | segment a hex 0 |
| Hex0[1] | F22 | segment b hex 0 |
| Hex0[2] | E17 | segment c hex 0 |
| Hex0[3] | L26 | segment d hex 0 |
| Hex0[4] | L25 | segment e hex 0 |
| Hex0[5] | J22 | segment f hex 0 |
| Hex0[6] | H22 | segment g hex 0 |
| Hex1[0] | M24 | segment a hex 1 |
| Hex1[1] | Y22 | segment b hex 1 |
| Hex1[2] | W21 | segment c hex 1 |
| Hex1[3] | W22 | segment d hex 1 |
| Hex1[4] | W25 | segment e hex 1 |
| Hex1[5] | U23 | segment f hex 1 |
| Hex1[6] | U24 | segment g hex 1 |
| Hex2[0] | AA25 | segment a hex 2 |
| Hex2[1] | AA26 | segment b hex 2 |
| Hex2[2] | Y25 | segment c hex 2 |
| Hex2[3] | W26 | segment d hex 2 |
| Hex2[4] | Y26 | segment e hex 2 |
| Hex2[5] | W27 | segment f hex 2 |
| Hex2[6] | W28 | segment g hex 2 |
| Hex3[0] | V21 | segment a hex 3 |
| Hex3[1] | U21 | segment b hex 3 |
| Hex3[2] | AB20 | segment c hex 3 |
| Hex3[3] | AA21 | segment d hex 3 |
| Hex3[4] | AD24 | segment e hex 3 |
| Hex3[5] | AF23 | segment f hex 3 |
| Hex3[6] | Y19 | segment g hex 3 |
| Hex4[0] | AB19 | segment a hex 4 |
| Hex4[1] | AA19 | segment b hex 4 |
| Hex4[2] | AG21 | segment c hex 4 |
| Hex4[3] | AH21 | segment d hex 4 |
| Hex4[4] | AE19 | segment e hex 4 |
| Hex4[5] | AF19 | segment f hex 4 |
| Hex4[6] | AE18 | segment g hex 4 |
| Hex5[0] | AD18 | segment a hex 5 |
| Hex5[1] | AC18 | segment b hex 5 |
| Hex5[2] | AB18 | segment c hex 5 |
| Hex5[3] | AH19 | segment d hex 5 |
| Hex5[4] | AG19 | segment e hex 5 |
| Hex5[5] | AF18 | segment f hex 5 |
| Hex5[6] | AH18 | segment g hex 5 |
| Hex6[0] | AA17 | segment a hex 6 |
| Hex6[1] | AB16 | segment b hex 6 |
| Hex6[2] | AA16 | segment c hex 6 |
| Hex6[3] | AB17 | segment d hex 6 |
| Hex6[4] | AB15 | segment e hex 6 |
| Hex6[5] | AA15 | segment f hex 6 |
| Hex6[6] | AC17 | segment g hex 6 |
| Hex7[0] | AD17 | segment a hex 7 |
| Hex7[1] | AE17 | segment b hex 7 |
| Hex7[2] | AG17 | segment c hex 7 |
| Hex7[3] | AH17 | segment d hex 7 |
| Hex7[4] | AF17 | segment e hex 7 |
| Hex7[5] | AG18 | segment f hex 7 |
| Hex7[6] | AA14 | segment g hex 7 |

fichier .csv de la carte altera complète *Attention, les noms des signaux d'entrée/sortie sont ceux de la carte, que l'on retrouve dans les tableaux ci-dessus. Il faudra adapter ces noms à ceux de votre*

application

Quartus II Version 5.1 Internal Build 160 09/19/2005 TO Full Version,
File: D:\de2_pins\de2_pins.csv,
Generated on: Wed Sep 28 09:40:34 2005,
Note: The column header names should not be changed if you wish to import this .csv file into the Quartus II software.,

To,Location

SW[0],PIN_N25
SW[1],PIN_N26
SW[2],PIN_P25
SW[3],PIN_AE14
SW[4],PIN_AF14
SW[5],PIN_AD13
SW[6],PIN_AC13
SW[7],PIN_C13
SW[8],PIN_B13
SW[9],PIN_A13
SW[10],PIN_N1
SW[11],PIN_P1
SW[12],PIN_P2
SW[13],PIN_T7
SW[14],PIN_U3
SW[15],PIN_U4
SW[16],PIN_V1
SW[17],PIN_V2
DRAM_ADDR[0],PIN_T6
DRAM_ADDR[1],PIN_V4
DRAM_ADDR[2],PIN_V3
DRAM_ADDR[3],PIN_W2
DRAM_ADDR[4],PIN_W1
DRAM_ADDR[5],PIN_U6
DRAM_ADDR[6],PIN_U7
DRAM_ADDR[7],PIN_U5
DRAM_ADDR[8],PIN_W4
DRAM_ADDR[9],PIN_W3
DRAM_ADDR[10],PIN_Y1
DRAM_ADDR[11],PIN_V5
DRAM_BA_0,PIN_AE2
DRAM_BA_1,PIN_AE3

DRAM_CAS_N,PIN_AB3
DRAM_CKE,PIN_AA6
DRAM_CLK,PIN_AA7
DRAM_CS_N,PIN_AC3
DRAM_DQ[0],PIN_V6
DRAM_DQ[1],PIN_AA2
DRAM_DQ[2],PIN_AA1
DRAM_DQ[3],PIN_Y3
DRAM_DQ[4],PIN_Y4
DRAM_DQ[5],PIN_R8
DRAM_DQ[6],PIN_T8
DRAM_DQ[7],PIN_V7
DRAM_DQ[8],PIN_W6
DRAM_DQ[9],PIN_AB2
DRAM_DQ[10],PIN_AB1
DRAM_DQ[11],PIN_AA4
DRAM_DQ[12],PIN_AA3
DRAM_DQ[13],PIN_AC2
DRAM_DQ[14],PIN_AC1
DRAM_DQ[15],PIN_AA5
DRAM_LDQM,PIN_AD2
DRAM_UDQM,PIN_Y5
DRAM_RAS_N,PIN_AB4
DRAM_WE_N,PIN_AD3
FL_ADDR[0],PIN_AC18
FL_ADDR[1],PIN_AB18
FL_ADDR[2],PIN_AE19
FL_ADDR[3],PIN_AF19
FL_ADDR[4],PIN_AE18
FL_ADDR[5],PIN_AF18
FL_ADDR[6],PIN_Y16
FL_ADDR[7],PIN_AA16
FL_ADDR[8],PIN_AD17
FL_ADDR[9],PIN_AC17
FL_ADDR[10],PIN_AE17
FL_ADDR[11],PIN_AF17
FL_ADDR[12],PIN_W16
FL_ADDR[13],PIN_W15
FL_ADDR[14],PIN_AC16

FL_ADDR[15],PIN_AD16
FL_ADDR[16],PIN_AE16
FL_ADDR[17],PIN_AC15
FL_ADDR[18],PIN_AB15
FL_ADDR[19],PIN_AA15
FL_ADDR[20],PIN_Y15
FL_ADDR[21],PIN_Y14
FL_CE_N,PIN_V17
FL_OE_N,PIN_W17
FL_DQ[0],PIN_AD19
FL_DQ[1],PIN_AC19
FL_DQ[2],PIN_AF20
FL_DQ[3],PIN_AE20
FL_DQ[4],PIN_AB20
FL_DQ[5],PIN_AC20
FL_DQ[6],PIN_AF21
FL_DQ[7],PIN_AE21
FL_RST_N,PIN_AA18
FL_WE_N,PIN_AA17
HEX0[0],PIN_AF10
HEX0[1],PIN_AB12
HEX0[2],PIN_AC12
HEX0[3],PIN_AD11
HEX0[4],PIN_AE11
HEX0[5],PIN_V14
HEX0[6],PIN_V13
HEX1[0],PIN_V20
HEX1[1],PIN_V21
HEX1[2],PIN_W21
HEX1[3],PIN_Y22
HEX1[4],PIN_AA24
HEX1[5],PIN_AA23
HEX1[6],PIN_AB24
HEX2[0],PIN_AB23
HEX2[1],PIN_V22
HEX2[2],PIN_AC25
HEX2[3],PIN_AC26
HEX2[4],PIN_AB26
HEX2[5],PIN_AB25

HEX 2[6],PIN _ Y24
HEX 3[0],PIN _ Y23
HEX 3[1],PIN _ AA 25
HEX 3[2],PIN _ AA 26
HEX 3[3],PIN _ Y26
HEX 3[4],PIN _ Y25
HEX 3[5],PIN _ U22
HEX 3[6],PIN _ W24
HEX 4[0],PIN _ U9
HEX 4[1],PIN _ U1
HEX 4[2],PIN _ U2
HEX 4[3],PIN _ T4
HEX 4[4],PIN _ R7
HEX 4[5],PIN _ R6
HEX 4[6],PIN _ T3
HEX 5[0],PIN _ T2
HEX 5[1],PIN _ P6
HEX 5[2],PIN _ P7
HEX 5[3],PIN _ T9
HEX 5[4],PIN _ R5
HEX 5[5],PIN _ R4
HEX 5[6],PIN _ R3
HEX 6[0],PIN _ R2
HEX 6[1],PIN _ P4
HEX 6[2],PIN _ P3
HEX 6[3],PIN _ M2
HEX 6[4],PIN _ M3
HEX 6[5],PIN _ M5
HEX 6[6],PIN _ M4
HEX 7[0],PIN _ L3
HEX 7[1],PIN _ L2
HEX 7[2],PIN _ L9
HEX 7[3],PIN _ L6
HEX 7[4],PIN _ L7
HEX 7[5],PIN _ P9
HEX 7[6],PIN _ N9
KEY[0],PIN _ G26
KEY[1],PIN _ N23
KEY[2],PIN _ P23

KEY[3],PIN_W26
LEDR[0],PIN_AE23
LEDR[1],PIN_AF23
LEDR[2],PIN_AB21
LEDR[3],PIN_AC22
LEDR[4],PIN_AD22
LEDR[5],PIN_AD23
LEDR[6],PIN_AD21
LEDR[7],PIN_AC21
LEDR[8],PIN_AA14
LEDR[9],PIN_Y13
LEDR[10],PIN_AA13
LEDR[11],PIN_AC14
LEDR[12],PIN_AD15
LEDR[13],PIN_AE15
LEDR[14],PIN_AF13
LEDR[15],PIN_AE13
LEDR[16],PIN_AE12
LEDR[17],PIN_AD12
LEDG[0],PIN_AE22
LEDG[1],PIN_AF22
LEDG[2],PIN_W19
LEDG[3],PIN_V18
LEDG[4],PIN_U18
LEDG[5],PIN_U17
LEDG[6],PIN_AA20
LEDG[7],PIN_Y18
LEDG[8],PIN_Y12
CLOCK_27,PIN_D13
CLOCK_50,PIN_N2
EXT_CLOCK,PIN_P26
PS2_CLK,PIN_D26
PS2_DAT,PIN_C24
UART_RXD,PIN_C25
UART_TXD,PIN_B25
LCD_RW,PIN_K4
LCD_EN,PIN_K3
LCD_RS,PIN_K1
LCD_DATA[0],PIN_J1

LCD_DATA[1],PIN_J2
LCD_DATA[2],PIN_H1
LCD_DATA[3],PIN_H2
LCD_DATA[4],PIN_J4
LCD_DATA[5],PIN_J3
LCD_DATA[6],PIN_H4
LCD_DATA[7],PIN_H3
LCD_ON,PIN_L4
LCD_BLON,PIN_K2
SRAM_ADDR[0],PIN_AE4
SRAM_ADDR[1],PIN_AF4
SRAM_ADDR[2],PIN_AC5
SRAM_ADDR[3],PIN_AC6
SRAM_ADDR[4],PIN_AD4
SRAM_ADDR[5],PIN_AD5
SRAM_ADDR[6],PIN_AE5
SRAM_ADDR[7],PIN_AF5
SRAM_ADDR[8],PIN_AD6
SRAM_ADDR[9],PIN_AD7
SRAM_ADDR[10],PIN_V10
SRAM_ADDR[11],PIN_V9
SRAM_ADDR[12],PIN_AC7
SRAM_ADDR[13],PIN_W8
SRAM_ADDR[14],PIN_W10
SRAM_ADDR[15],PIN_Y10
SRAM_ADDR[16],PIN_AB8
SRAM_ADDR[17],PIN_AC8
SRAM_DQ[0],PIN_AD8
SRAM_DQ[1],PIN_AE6
SRAM_DQ[2],PIN_AF6
SRAM_DQ[3],PIN_AA9
SRAM_DQ[4],PIN_AA10
SRAM_DQ[5],PIN_AB10
SRAM_DQ[6],PIN_AA11
SRAM_DQ[7],PIN_Y11
SRAM_DQ[8],PIN_AE7
SRAM_DQ[9],PIN_AF7
SRAM_DQ[10],PIN_AE8
SRAM_DQ[11],PIN_AF8

SRAM_DQ[12],PIN_W11
SRAM_DQ[13],PIN_W12
SRAM_DQ[14],PIN_AC9
SRAM_DQ[15],PIN_AC10
SRAM_WE_N,PIN_AE10
SRAM_OE_N,PIN_AD10
SRAM_UB_N,PIN_AF9
SRAM_LB_N,PIN_AE9
SRAM_CE_N,PIN_AC11
OTG_ADDR[0],PIN_K7
OTG_ADDR[1],PIN_F2
OTG_CS_N,PIN_F1
OTG_RD_N,PIN_G2
OTG_WR_N,PIN_G1
OTG_RST_N,PIN_G5
OTG_DATA[0],PIN_F4
OTG_DATA[1],PIN_D2
OTG_DATA[2],PIN_D1
OTG_DATA[3],PIN_F7
OTG_DATA[4],PIN_J5
OTG_DATA[5],PIN_J8
OTG_DATA[6],PIN_J7
OTG_DATA[7],PIN_H6
OTG_DATA[8],PIN_E2
OTG_DATA[9],PIN_E1
OTG_DATA[10],PIN_K6
OTG_DATA[11],PIN_K5
OTG_DATA[12],PIN_G4
OTG_DATA[13],PIN_G3
OTG_DATA[14],PIN_J6
OTG_DATA[15],PIN_K8
OTG_INT0,PIN_B3
OTG_INT1,PIN_C3
OTG_DACK0_N,PIN_C2
OTG_DACK1_N,PIN_B2
OTG_DREQ0,PIN_F6
OTG_DREQ1,PIN_E5
OTG_FSPEED,PIN_F3
OTG_LSPEED,PIN_G6

TDI,PIN_B14
TCS,PIN_A14
TCK,PIN_D14
TDO,PIN_F14
TD_RESET,PIN_C4
VGA_R[0],PIN_C8
VGA_R[1],PIN_F10
VGA_R[2],PIN_G10
VGA_R[3],PIN_D9
VGA_R[4],PIN_C9
VGA_R[5],PIN_A8
VGA_R[6],PIN_H11
VGA_R[7],PIN_H12
VGA_R[8],PIN_F11
VGA_R[9],PIN_E10
VGA_G[0],PIN_B9
VGA_G[1],PIN_A9
VGA_G[2],PIN_C10
VGA_G[3],PIN_D10
VGA_G[4],PIN_B10
VGA_G[5],PIN_A10
VGA_G[6],PIN_G11
VGA_G[7],PIN_D11
VGA_G[8],PIN_E12
VGA_G[9],PIN_D12
VGA_B[0],PIN_J13
VGA_B[1],PIN_J14
VGA_B[2],PIN_F12
VGA_B[3],PIN_G12
VGA_B[4],PIN_J10
VGA_B[5],PIN_J11
VGA_B[6],PIN_C11
VGA_B[7],PIN_B11
VGA_B[8],PIN_C12
VGA_B[9],PIN_B12
VGA_CLK,PIN_B8
VGA_BLANK,PIN_D6
VGA_HS,PIN_A7
VGA_VS,PIN_D8

VGA_SYNC,PIN_B7
I2C_SCLK,PIN_A6
I2C_SDAT,PIN_B6
TD_DATA[0],PIN_J9
TD_DATA[1],PIN_E8
TD_DATA[2],PIN_H8
TD_DATA[3],PIN_H10
TD_DATA[4],PIN_G9
TD_DATA[5],PIN_F9
TD_DATA[6],PIN_D7
TD_DATA[7],PIN_C7
TD_HS,PIN_D5
TD_VS,PIN_K9
AUD_ADCLRCK,PIN_C5
AUD_ADCDAT,PIN_B5
AUD_DACLRCK,PIN_C6
AUD_DACDAT,PIN_A4
AUD_XCK,PIN_A5
AUD_BCLK,PIN_B4
ENET_DATA[0],PIN_D17
ENET_DATA[1],PIN_C17
ENET_DATA[2],PIN_B18
ENET_DATA[3],PIN_A18
ENET_DATA[4],PIN_B17
ENET_DATA[5],PIN_A17
ENET_DATA[6],PIN_B16
ENET_DATA[7],PIN_B15
ENET_DATA[8],PIN_B20
ENET_DATA[9],PIN_A20
ENET_DATA[10],PIN_C19
ENET_DATA[11],PIN_D19
ENET_DATA[12],PIN_B19
ENET_DATA[13],PIN_A19
ENET_DATA[14],PIN_E18
ENET_DATA[15],PIN_D18
ENET_CLK,PIN_B24
ENET_CMD,PIN_A21
ENET_CS_N,PIN_A23
ENET_INT,PIN_B21

ENET_RD_N,PIN_A22
ENET_WR_N,PIN_B22
ENET_RST_N,PIN_B23
IRDA_TXD,PIN_AE24
IRDA_RXD,PIN_AE25
SD_DAT,PIN_AD24
SD_DAT3,PIN_AC23
SD_CMD,PIN_Y21
SD_CLK,PIN_AD25
GPIO_0[0],PIN_D25
GPIO_0[1],PIN_J22
GPIO_0[2],PIN_E26
GPIO_0[3],PIN_E25
GPIO_0[4],PIN_F24
GPIO_0[5],PIN_F23
GPIO_0[6],PIN_J21
GPIO_0[7],PIN_J20
GPIO_0[8],PIN_F25
GPIO_0[9],PIN_F26
GPIO_0[10],PIN_N18
GPIO_0[11],PIN_P18
GPIO_0[12],PIN_G23
GPIO_0[13],PIN_G24
GPIO_0[14],PIN_K22
GPIO_0[15],PIN_G25
GPIO_0[16],PIN_H23
GPIO_0[17],PIN_H24
GPIO_0[18],PIN_J23
GPIO_0[19],PIN_J24
GPIO_0[20],PIN_H25
GPIO_0[21],PIN_H26
GPIO_0[22],PIN_H19
GPIO_0[23],PIN_K18
GPIO_0[24],PIN_K19
GPIO_0[25],PIN_K21
GPIO_0[26],PIN_K23
GPIO_0[27],PIN_K24
GPIO_0[28],PIN_L21
GPIO_0[29],PIN_L20

GPIO_0[30],PIN_J25
GPIO_0[31],PIN_J26
GPIO_0[32],PIN_L23
GPIO_0[33],PIN_L24
GPIO_0[34],PIN_L25
GPIO_0[35],PIN_L19
GPIO_1[0],PIN_K25
GPIO_1[1],PIN_K26
GPIO_1[2],PIN_M22
GPIO_1[3],PIN_M23
GPIO_1[4],PIN_M19
GPIO_1[5],PIN_M20
GPIO_1[6],PIN_N20
GPIO_1[7],PIN_M21
GPIO_1[8],PIN_M24
GPIO_1[9],PIN_M25
GPIO_1[10],PIN_N24
GPIO_1[11],PIN_P24
GPIO_1[12],PIN_R25
GPIO_1[13],PIN_R24
GPIO_1[14],PIN_R20
GPIO_1[15],PIN_T22
GPIO_1[16],PIN_T23
GPIO_1[17],PIN_T24
GPIO_1[18],PIN_T25
GPIO_1[19],PIN_T18
GPIO_1[20],PIN_T21
GPIO_1[21],PIN_T20
GPIO_1[22],PIN_U26
GPIO_1[23],PIN_U25
GPIO_1[24],PIN_U23
GPIO_1[25],PIN_U24
GPIO_1[26],PIN_R19
GPIO_1[27],PIN_T19
GPIO_1[28],PIN_U20
GPIO_1[29],PIN_U21
GPIO_1[30],PIN_V26
GPIO_1[31],PIN_V25
GPIO_1[32],PIN_V24
GPIO_1[33],PIN_V23
GPIO_1[34],PIN_W25
GPIO_1[35],PIN_W23