ΕΡΓΑΣΤΗΡΙΟ ΨΗΦΙΑΚΗΣ ΕΠΕΞΕΡΓΑΣΙΑΣ ΣΗΜΑΤΩΝ

## Εργαστήριο 4

## Α/D, D/A, Δειγματοληψία και FIR φίλτρα

|  |
| --- |
| Ομάδα 17 9-11-2018 |
| Ασημακόπουλος Κωνσταντίνος 1046966 |
| Λουκαρέας Παύλος 1046970 |

**Περίληψη**

Σε αυτή την εργαστηριακή άσκηση εξοικειωθήκαμε με τη ρύθμιση των περιφερειακών για τη μετατροπή A/D και D/A στην πλακέτα για την εκτέλεση διαφόρων λειτουργιών. Ακόμη στα επόμενα ερωτήματα υλοποιήσαμε ένα ψηφιακό φίλτρο με τη χρήση του κώδικα της εργαστηριακής άσκησης 2 όπου γίνεται η υλοποίηση του εσωτερικού γινομένου δύο διανυσμάτων.

**Άσκηση 4.1**

Σε αυτό το ερώτημα εκτελούμε όλα τα βήματα που απαιτούνται από την εκφώνηση για τη ρύθμιση της σειριακής θύρας McBSP1. Ο κώδικας παρακάτω υλοποιεί την αρχικοποίηση.

RCR .set 0x190000C

XCR .set 0x1900010

enable\_value .set 0x000000A0

.def \_entry

.text

\_entry: MVKL .S1 RCR, A0

MVKH .S1 RCR, A0

MVKL .S1 XCR, A1

MVKH .S1 XCR, A1

MVKL .S1 enable\_value, A2

MVKH .S1 enable\_value, A2

STW .D1 A2, \*A0

STW .D1 A2, \*A1

.end

Όπως φαίνεται φορτώνοντας την enable\_value σε αυτούς τους δύο καταχωρητές μνήμης ρυθμίζουμε το κανάλι να δέχεται και να μεταδίδει 32 bit σε κάθε φάση.

**Άσκηση 4.2**

Σε αυτό το ερώτημα θέλουμε να μεταδώσουμε τα δεδομένα του καταχωρητή Α0 στη σειριακή έξοδο. Για να το κάνουμε αυτό θα πρέπει να γράψουμε τα δεδομένα στον καταχωρητή DXR. Πριν το κάνουμε αυτό θα πρέπει να ελέγξουμε αν είναι κενός. Αυτό μπορούμε να το δούμε ελέγχοντας το 17ο bit του καταχωρητή SPCR. Αν αυτό γίνει 1 τότε μπορούμε να γράψουμε στον DXR. Ο κώδικας που υλοποιεί τη ρουτίνα είναι ο παρακάτω.

SPCR .set 0x1900008

DXR .set 0x1900004

.def entry

.text

entry: MVKL .S1 DXR,A1

MVKH .S1 DXR,A1

MVKL .S1 SPCR, A2

MVKH .S1 SPCR, A2

MVKL .S1 0x10000, A3

MVKH .S1 0x10000, A3

loop: LDW \*A2, B1

NOP 4

AND B1, A3, B1 ;Isolating 17th position

XOR B1, A3, B1 ;Xor logic for inverting

[B1] B loop

NOP 5

STW A0, \*A1 ;Storing input

IDLE

.end

**Άσκηση 4.3**

Σε αυτό το ερώτημα θέλουμε να υλοποιήσουμε την αντίστροφη λειτουργία και να διαβάσουμε τα δεδομένα από τη σειριακή είσοδο. Η λογική είναι παρόμοια, για να διαβάσουμε τα δεδομένα ο καταχωρητής DRR πρέπει να είναι έτοιμος για ανάγνωση. Η μόνη διαφορά είναι ότι για να διαβάσουμε τα δεδομένα από τον καταχωρητή πρώτα ελέγχουμε το πρώτο bit του SPCR.

SPCR .set 0x1900008

DRR .set 0x1900000

.def entry

.text

entry: MVKL .S1 SPCR, A1

MVKH .S1 SPCR, A1

MVKL .S1 DRR, A2

MVKH .S1 DRR, A2

MVK .S2 0x1, B1

loop: LDW \*A1, B0

NOP 4

AND B0, B1, B0 ;Isolating first bit

XOR B0, B1, B0 ;Condition for exiting loop

[B0] B loop

NOP 5

LDW \*A2, A3 ;Reading input

IDLE

.end

**Άσκηση 4.4**

Σε αυτό το ερώτημα συνδέουμε στην πλακέτα μια πηγή εισόδου ήχου και σαν έξοδο κάποια ηχεία. Ο στόχος μας είναι ότι είσοδο παίρνουμε να την αναπαράγουμε στα ηχεία. Για να το πετύχουμε αυτό θα συνδυάσουμε τις δύο παραπάνω λειτουργίες σε μία. Ο κώδικας μας είναι ο παρακάτω.

DRR .set 0x1900000

DXR .set 0x1900004

SPCR .set 0x1900008

.def \_entry

.text

\_entry: MVKL .S1 DRR, A0

MVKH .S1 DRR, A0

MVKL .S1 DXR, A1

MVKH .S1 DXR, A1

MVKL .S1 SPCR, A2

MVKH .S1 SPCR, A2

MVKL .S1 0x10000, A3

MVKH .S1 0x10000, A3

MVK .S2 0x1, B2

loopa: LDW \*A2, B0

NOP 4

AND B0, B2, B0 ;Isolating first bit

XOR B0, B2, B0 ;Xor logic for inverting

[B0] B loopa

NOP 5

LDW \*A0, A4 ;Reading input

NOP 4

loopb: LDW \*A2, B1

NOP 4

AND B1, A3, B1 ;Isolating 17th bit

XOR B1, A3, B1 ;Xor logic for inverting

[B1] B loopb

NOP 5

STW .D1 A4, \*A1 ;Storing input

B loopa

NOP 5

IDLE

.end

**Άσκηση 4.5**

Σε αυτό το ερώτημα θέλουμε να κάνουμε ακριβώς την ίδια διαδικασία με το προηγούμενο ερώτημα με τη μόνη διαφορά ότι αντί να ελέγχουμε συνεχώς τα bits του SPCR περιμένουμε τις διακοπές που γίνονται όταν τα bits που μας ενδιαφέρουν γίνονται 1. Επειδή οι μετατροπές A/D και D/A εμφανίζονται με υψηλό ρυθμό (48 kHz) αρκεί να εξυπηρετήσουμε μόνο μία διακοπή από τις δύο (πχ RINT1) για να λάβουμε και στη συνέχεια να αποστείλουμε ψηφιακό σήμα. Ο κώδικας μας είναι ο παρακάτω:

DRR .set 0x1900000

DXR .set 0x1900004

SPCR .set 0x1900008

RCR .set 0x190000C

XCR .set 0x1900010

enable\_value .set 0x000000A0

IMH\_VALUE .set 0xF

IMH\_pointer .set 0x019c0000

.def INT4\_R

.def \_entry

.text

\_entry: ΜVKL .S1 RCR, A0

MVKH .S1 RCR, A0

MVKL .S1 XCR, A1

MVKH .S1 XCR, A1

MVKL .S1 enable\_value, A2

MVKH .S1 enable\_value, A2

STW .D1 A2, \*A0

STW .D1 A2, \*A1

MVKL .S1 IMH\_pointer, A5

MVKH .S1 IMH\_pointer, A5

MVKL .S1 IMH\_VALUE, A6

MVKH .S1 IMH\_VALUE, A6

STW .D1 A6, \*A5 ;load the interrupts in IML

MVKL .S1 DRR, A0

MVKH .S1 DRR, A0

MVKL .S1 DXR, A1

MVKH .S1 DXR, A1

MVKL .S1 SPCR, A2

MVKH .S1 SPCR, A2

MVKL .S1 0x10000, A3

MVKH .S1 0x10000, A3

MVK .S2 0x1, B2

;enable the interrupts

MVKL .S2 0X403, B0 ;First bit of NMI and INT10

MVC .S2 B0, IER

MVKL .S2 0x1, B1 ;First bit of GIE

NOP

loopa: B loopa

MVC .S2 B1, CSR

NOP 4

INT4\_R: LDW \*A0, A4 ;Reading input

NOP 4

STW .D1 A4, \*A1 ;Storing input

B loopa

NOP 5

IDLE

.end

**Άσκηση 4.6**

Σε αυτό το ερώτημα θέλουμε να αποθηκεύσουμε τα δείγματα που παίρνουμε από το μικρόφωνο, να χωρίσουμε τα δείγματα σε αριστερό και δεξί κανάλι και να τα αποθηκεύσουμε σε buffers. Έπειτα θα αναπαραστήσουμε γραφικά στο πεδίο του χρόνου και της συχνότητας. Αρχικά δημιουργούμε δύο buffers των 256 halfwords και αρχίζουμε να αποθηκεύουμε τα δεδομένα εκεί. Μόλις αυτοί γεμίσουν αρχίζουμε και γράφουμε ξανά από την κορυφή τους. Σύμφωνα με τις οδηγίες εφαρμόζουμε ένα ημιτονικό σήμα των 12 KHz και σχεδιάζουμε τις γραφικές παραστάσεις. Ο κώδικας είναι:

DRR .set 0x1900000

DXR .set 0x1900004

SPCR .set 0x1900008

RCR .set 0x190000C

XCR .set 0x1900010

enable\_value .set 0x000000A0

IMH\_VALUE .set 0xF

IMH\_pointer .set 0x019c0000

.def INT4\_R

.def \_entry

.text

buffer\_left .space 512

buffer\_right .space 512

\_entry: MVKL .S1 buffer\_right, A8

MVKH .S1 buffer\_right, A8

ΜVKL .S1 buffer\_left, A9

MVKH .S1 buffer\_left, A9

MVKL .S1 RCR, A0

MVKH .S1 RCR, A0

MVKL .S1 XCR, A1

MVKH .S1 XCR, A1

MVKL .S1 enable\_value, A2

MVKH .S1 enable\_value, A2

STW .D1 A2, \*A0

STW .D1 A2, \*A1

MVKL .S1 IMH\_pointer, A5

MVKH .S1 IMH\_pointer, A5

MVKL .S1 IMH\_VALUE, A6

MVKH .S1 IMH\_VALUE, A6

STW .D1 A6, \*A5 ; load the interrupts in IML

MVKL .S1 DRR, A10

MVKH .S1 DRR, A10

MVKL .S1 DXR, A1

MVKH .S1 DXR, A1

MVKL .S1 SPCR, A2

MVKH .S1 SPCR, A2

MVKL .S1 0x10000, A3

MVKH .S1 0x10000, A3

MVK .S2 0x1, B2

;enable the interrupts

MVKL .S2 0X403, B0 ;First bit of NMI and INT10

MVC .S2 B0, IER

MVKL .S2 0x1, B2 ;First bit of GIE

NOP

loopb: MVKL .S2 0x1FF, B1

MVKH .S2 0x1FF, B1

loopa: B loopa

MVC .S2 B2, CSR

NOP 4

INT4\_R: LDW \*A10, A4 ;Reading input

NOP 4

MV .L1 A4, A11

STH .D1 A4, \*A8++

SHR .S1 A4,0x10, A7

STH .D1 A7,\*A9++

SUB .D2 B1, B2, B1

[!B1] B loopb

STW .D1 A11, \*A1 ;Storing input

B loopa

NOP 5

IDLE

.end

