ΔΗΜΟΚΡΙΤΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΡΑΚΗΣ

ΤΜΗΜΑ ΗΜ&ΜΥ

ΣΧΕΔΙΑΣΜΟΣ ΕΝΣΩΜΑΤΩΜΕΝΩΝ ΣΥΣΤΗΜΑΤΩΝ

Τελική εργασία – 1ο μέρος – 12/07/2025

Στοιχεία Φοιτητών 9ου Εξαμήνου

Βαϊντερλής Άγγελος, 58647
Κοκορότσικου Αγνή Ιωάννα, 58767
Ξανθόπουλος Ηλίας, 58545

GitHub Page

Timer_A HAL – Documentation | embedded-systems-design-Timer_A

Επιβλέπων Καθηγητής
Δρ. Ιωάννης Βούρκας

Περιεχόμενα

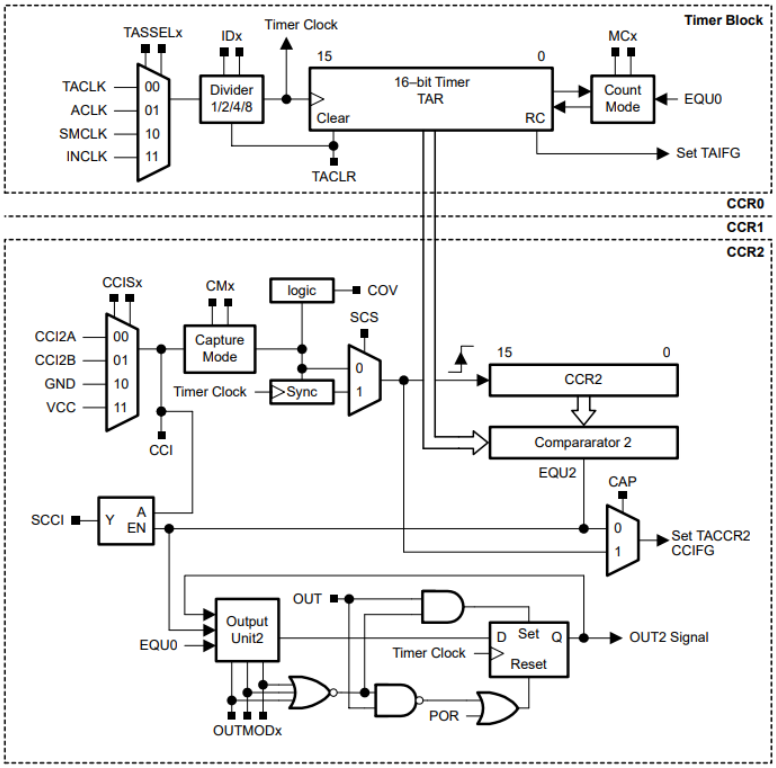
1. Εισαγωγή

Στο πλαίσιο του μαθήματος «Σχεδιασμός Ενσωματωμένων Συστημάτων» υλοποιήθηκε ένα Hardware Abstraction Layer (HAL) για το υποσύστημα Timer_A του μικροελεγκτή MSP430x1xx. Σκοπός της δοθείσας εργασίας είναι η απόκρυψη των χαμηλού επιπέδου λεπτομερειών διαμόρφωσης (configuration) του Timer_A πίσω από ένα σύνολο ορισμένων συναρτήσεων και δομών δεδομένων έτσι, ώστε η χρήση του χρονιστή να είναι πιο ασφαλής, ευανάγνωστη και επαναχρησιμοποιήσιμη από κώδικα εφαρμογών.

Η προσέγγιση ακολουθεί τη φιλοσοφία που παρουσιάστηκε στις εργαστηριακές ασκήσεις για το GPIO HAL, επεκτείνοντάς την στο module χρονομέτρησης. Η υλοποίηση βασίζεται στο κεφάλαιο 11 του επίσημου user manual των MSP430x1xx, το οποίο ορίζει τη δομή των καταχωρητών TACTL, TACCTLx, TAR και TACCRx και τις αντίστοιχες λειτουργικές δυνατότητες του Timer_A. (MSP430x1xx User's Guide, pp. 11.1-11.24)

2. Θεωρητικό υπόβαθρο

Ο Timer_A είναι ένας 16-bit timer/counter με τρεις capture/compare καταχωρητές (TACCR0 – TACCR2), όπως φαίνεται και στην παρακάτω Εικόνα 1. Υποστηρίζει τέσσερις λειτουργικές καταστάσεις (stop, up, continuous, up/down), δυνατότητα παραγωγής PWM σημάτων, καθώς και πολλαπλές πηγές διακοπών από τον ίδιο τον μετρητή ή από τα capture/compare blocks.



Εικόνα 1 – (MSP430x1xx User's Guide, p. 11.3)

Η βασική διαμόρφωση του Timer_A γίνεται μέσω του καταχωρητή ελέγχου TACTL. Τα bits TASSELx επιλέγουν την πηγή ρολογιού (TACLK, ACLK, SMCLK ή INCLK), τα bits IDx ορίζουν τον διαιρέτη εισόδου (διά 1, διά 2, διά 4 ή διά 8), ενώ τα bits MCx επιλέγουν τη λειτουργία (stop, up, continuous, up/down). Το bit TACLR επιτρέπει τον μηδενισμό του counter και του prescaler, ενώ τα TAIE και TAIFG ελέγχουν αντίστοιχα την ενεργοποίηση και την ένδειξη διακοπής υπερχείλισης του Timer_A.

Επίσης, ο καταχωρητής TAR περιέχει την τρέχουσα τιμή του 16-bit μετρητή. Η τιμή αυτή αυξάνεται ή μειώνεται ανάλογα με τον επιλεγμένο τρόπο λειτουργίας. (MSP430x1xx User's Guide, pp. 11.6-11.10)

Τέλος, κάθε capture/compare block διαθέτει έναν καταχωρητή ελέγχου TACCTLx και έναν καταχωρητή δεδομένων TACCRx. Τα bits OUTMODx του TACCTLx ορίζουν τη λειτουργία της εξόδου (set, reset, toggle) και χρησιμοποιούνται για την παραγωγή PWM κυματομορφών, ενώ το bit CCIE ενεργοποιεί τις διακοπές από το αντίστοιχο block. Ανάλογα με την τιμή του bit CAP, το block αυτό μπορεί να λειτουργήσει είτε σε compare mode (CAP = 0) για παραγωγή τακτικών συμβάντων/PWM, είτε σε capture mode (CAP = 1) για χρονοσήμανση εξωτερικών γεγονότων. (MSP430x1xx User's Guide, pp. 11.11-11.16)

3. Σχεδίαση του Timer_A HAL

Η σχεδίαση του HAL οργανώθηκε σε τρία επίπεδα, όπως έχει αναλυθεί και στα εργαστήρια: ορισμός ενδιάμεσων τύπων δεδομένων και enumerations, δομές παραμετροποίησης υψηλού επιπέδου και συναρτήσεις API που υλοποιούν τις αντίστοιχες λειτουργίες πάνω στους καταχωρητές του Timer_A.

3.1. Enumerations και bitfields

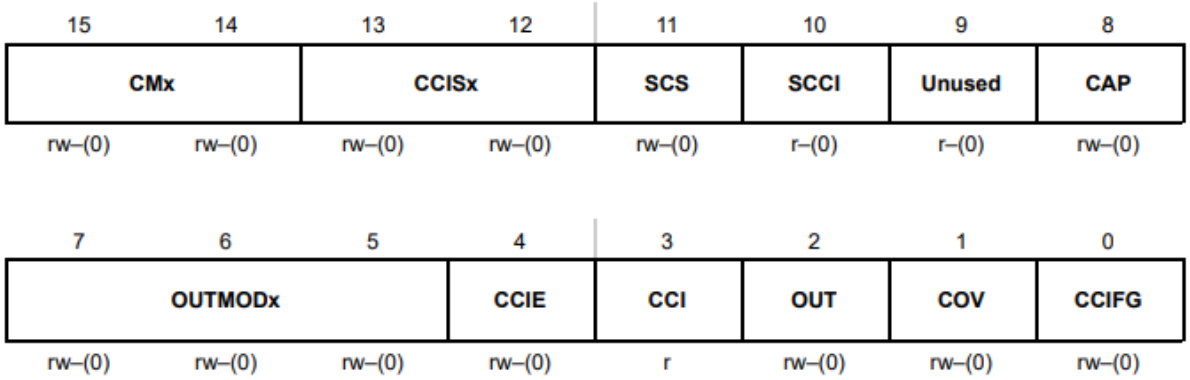
Στο header αρχείο timerA_HAL_19.h ορίζονται enumerations που αποτυπώνουν την αρχικοποίηση τιμής των πεδίων των καταχωρητών, όπως αυτά ορίζονται στο manual. Για παράδειγμα, οι τύποι TimerA_ClockSource, TimerA_ClockDivider και TimerA_Mode αντιστοιχούν στα bits TASSELx, IDx και MCx του TACTL. (MSP430x1xx User's Guide, pp. 11.20-11.23)

Register	Short Form	Register Type	Address	Initial State
Timer_A control	TACTL	Read/write	0160h	Reset with POR
Timer_A counter	TAR	Read/write	0170h	Reset with POR
Timer_A capture/compare control 0	TACCTL0	Read/write	0162h	Reset with POR
Timer_A capture/compare 0	TACCR0	Read/write	0172h	Reset with POR
Timer_A capture/compare control 1	TACCTL1	Read/write	0164h	Reset with POR
Timer_A capture/compare 1	TACCR1	Read/write	0174h	Reset with POR
Timer_A capture/compare control 2	TACCTL2	Read/write	0166h	Reset with POR
Timer_A capture/compare 2	TACCR2	Read/write	0176h	Reset with POR
Timer_A Interrupt Vector	TAIV	Read only	012Eh	Reset with POR

Εικόνα 2 – (MSP430x1xx User's Guide, p. 11.19)

Αντίστοιχα, ο τύπος TimerA_OutputMode αποτυπώνει τις οκτώ λειτουργίες εξόδου του Output Unit μέσω των τριών bits OUTMODx του TACCTLx, επιτρέποντας την επιλογή του κατάλληλου PWM mode. (MSP430x1xx User's Guide, p. 11.13) Επιπλέον, έχουν οριστεί τα ζητούμενα bitfield struct/union τύποι TimerA_TACTL και TimerA_TACCTL, οι οποίοι χαρτογραφούν τα επιμέρους bits των καταχωρητών TACTL και TACCTLx σε ονομαστικά πεδία.

Οι πραγματικοί control καταχωρητές του Timer_A (TACTL, TAR, TACCTL0–2, TACCR0–2) δηλώνονται στο HAL ως μεταβλητές τύπου bitfield union, με διευθύνσεις που λήφθηκαν από την Εικόνα 2, χωρίς τη χρήση του device header.



Εικόνα 3 – (MSP430x1xx User's Guide, p. 11.20)

3.2. Δομές παραμετροποίησης και συναρτήσεις API

Για τη διαμόρφωση του Timer_A, ορίζονται αρχικά οι τρεις παρακάτω βασικές δομές, οι οποίες χρησιμοποιούνται ως είσοδος στις συναρτήσεις του HAL, επιτρέποντας στον κώδικα εφαρμογής να περιγράφει το ζητούμενο configuration σε υψηλότερο επίπεδο αφαίρεσης, χωρίς άμεσες εγγραφές σε καταχωρητές.

Δομές	Περιγραφή
TimerA_Config	Περιγράφει το γενικό configuration του TACTL και του TACCR0, περιλαμβάνοντας πηγή ρολογιού, διαιρέτη, τρόπο λειτουργίας, περίοδο και ενεργοποίηση της διακοπής του Timer_A.
TimerA_ChannelConfig	Αναφέρεται σε ένα συγκεκριμένο capture/compare κανάλι (CCR0, CCR1 ή CCR2) και περιλαμβάνει τη λειτουργία εξόδου (OUTMOD), την τιμή compare (TACCRx) και την ενεργοποίηση της αντίστοιχης διακοπής (CCIE).
TimerA_PWMConfig	Εξειδίκευση για παραγωγή PWM, συνδυάζοντας περίοδο (TACCR0), duty cycle (TACCRx) και output mode.

Στη συνέχεια, οι συναρτήσεις TimerA_ApplyConfig, TimerA_ConfigureChannel και TimerA_ConfigPWM υλοποιούν τη χαρτογράφηση αυτών των παραμέτρων στα αντίστοιχα bits των TACTL, TACCTLx και TACCRx. Παρέχονται επίσης οι βοηθητικές συναρτήσεις TimerA_Start, TimerA_Stop, TimerA_GetCounter, TimerA_ResetCounter, TimerA_SetPeriod, TimerA_SetDuty, οι οποίες κρύβουν την άμεση πρόσβαση στους καταχωρητές από τον κώδικα της εφαρμογής.

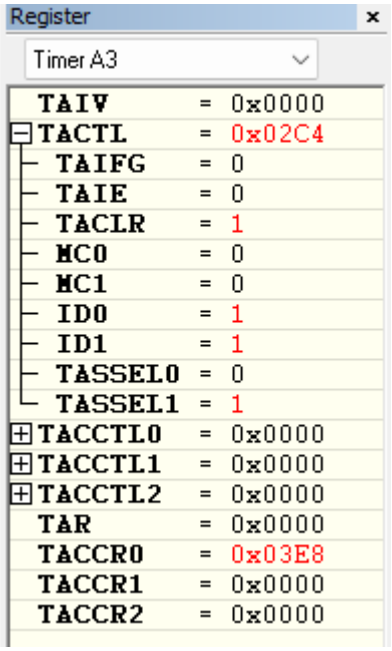
3.3. Αποτελέσματα συναρτήσεων

Παρακάτω φαίνονται τα απαραίτητα στιγμιότυπα οθόνης για την παρουσίαση των αποτελεσμάτων.

```
TimerA_Config basicCfg;

basicCfg.clockSource          = CLOCK_SMCLK;
basicCfg.clockDivider         = DIV_8;
basicCfg.mode                 = MODE_UP;
basicCfg.period               = 1000u;
basicCfg.enableTimerInterrupt = 0u;

// Apply TimerA configurations
TimerA_ApplyConfig(&basicCfg);
```

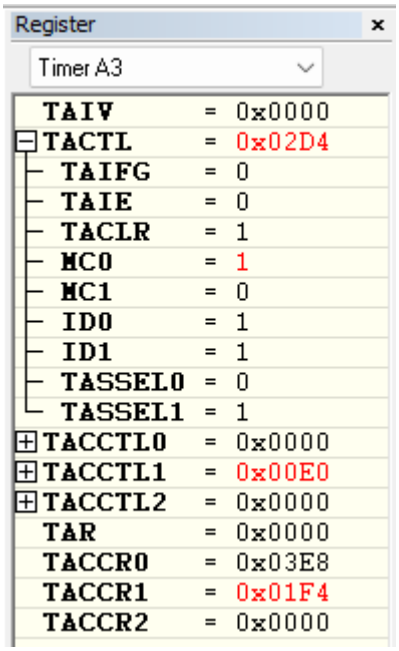


Εικόνα 4 – Χρήση της συνάρτησης ApplyConfig() και το αντίστοιχο αναμενόμενο αποτέλεσμα στους καταχωρητές του Timer_A

```
TimerA_PWMConfig pwmCfg;

pwmCfg.channel     = CHANNEL_1;
pwmCfg.period      = 1000u;
pwmCfg.duty        = 500u;
pwmCfg.outputMode = OUTMODE_RST_SET;

// Apply TimerA PWM configurations
TimerA_ConfigPWM(&pwmCfg); /* Immediate
```



Εικόνα 5 – Χρήση της συνάρτησης PWMConfig() και το αντίστοιχο αναμενόμενο αποτέλεσμα στον καταχωρητή TACCR1

```
// Change Duty Cycle in TACCR1 from 50% to 750/1000 = 75%
TimerA_SetDuty(CHANNEL_1, 750u);
```

```
TACCR1    = 0x02EE
```

Εικόνα 6 – Χρήση της συνάρτησης SetDuty() και το αντίστοιχο αναμενόμενο αποτέλεσμα στον καταχωρητή TACCR1

4. Βιβλιογραφία

[1] Instruments, T. (2003). *MSP430x1xx User's Guide*. Dallas, TX: Instruments, Texas.