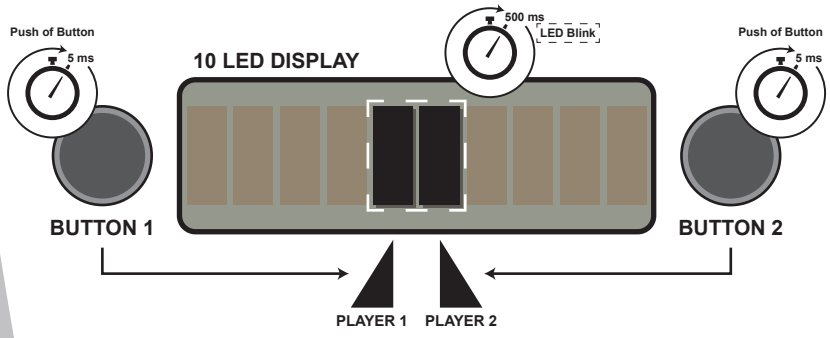
PROJECT

SUMO

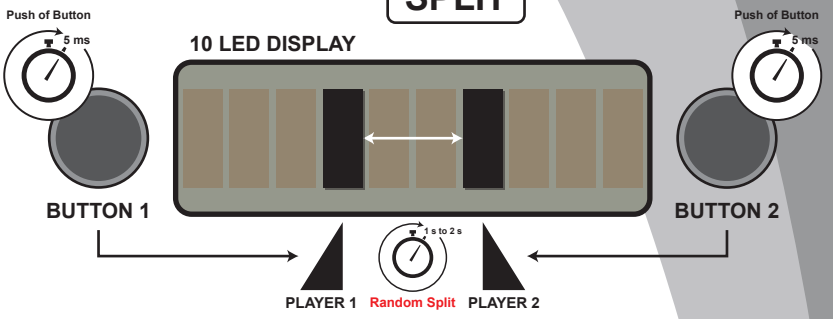
Program & Documention by Joel Meine

START



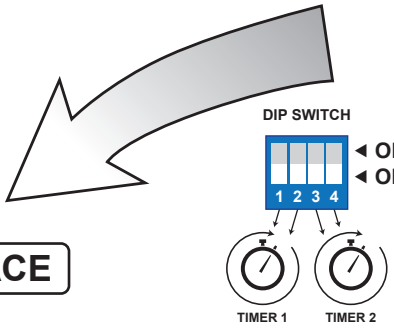
● ON ▶ Button Pushed ▶ SOLID LED
○ OFF ▶ Button not Pushed ▶ BLINK LED

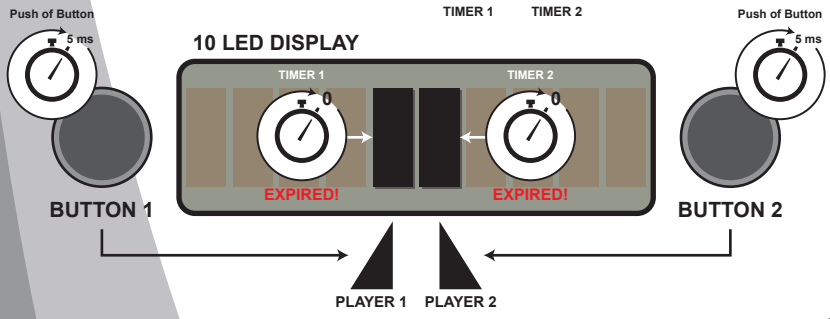
SPLIT



● ON ▶ Button Pushed ▶ START TIMER for Player
○ OFF ▶ Button not Pushed ▶ None

RACE

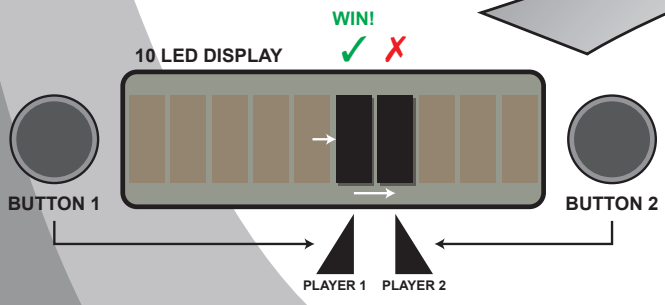




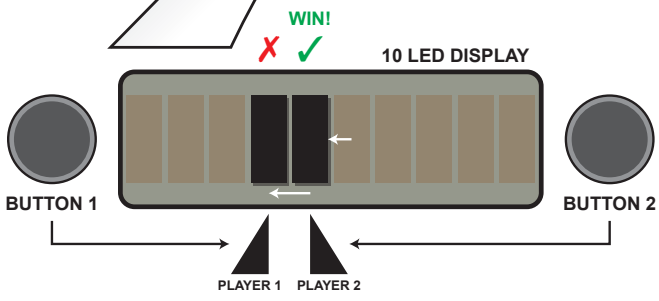
● ON ▶ Button Pushed ▶ Check if Player's TIMER Expired
○ OFF ▶ Button not Pushed ▶ None

Repeat from Present Positions until FINISH

RESULT

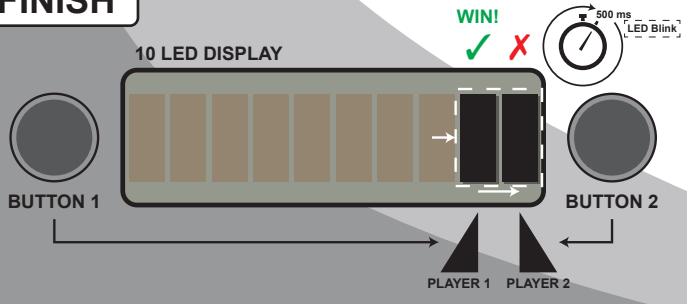


Player 1 Wins Round!

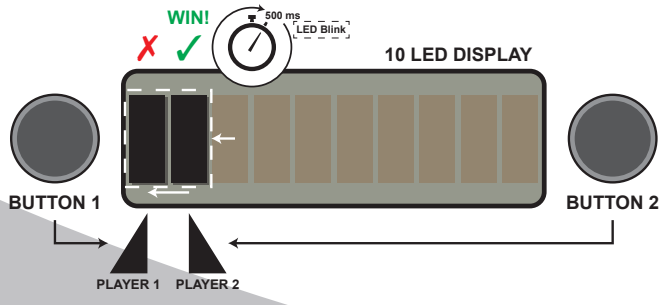


Player 2 Wins Round!

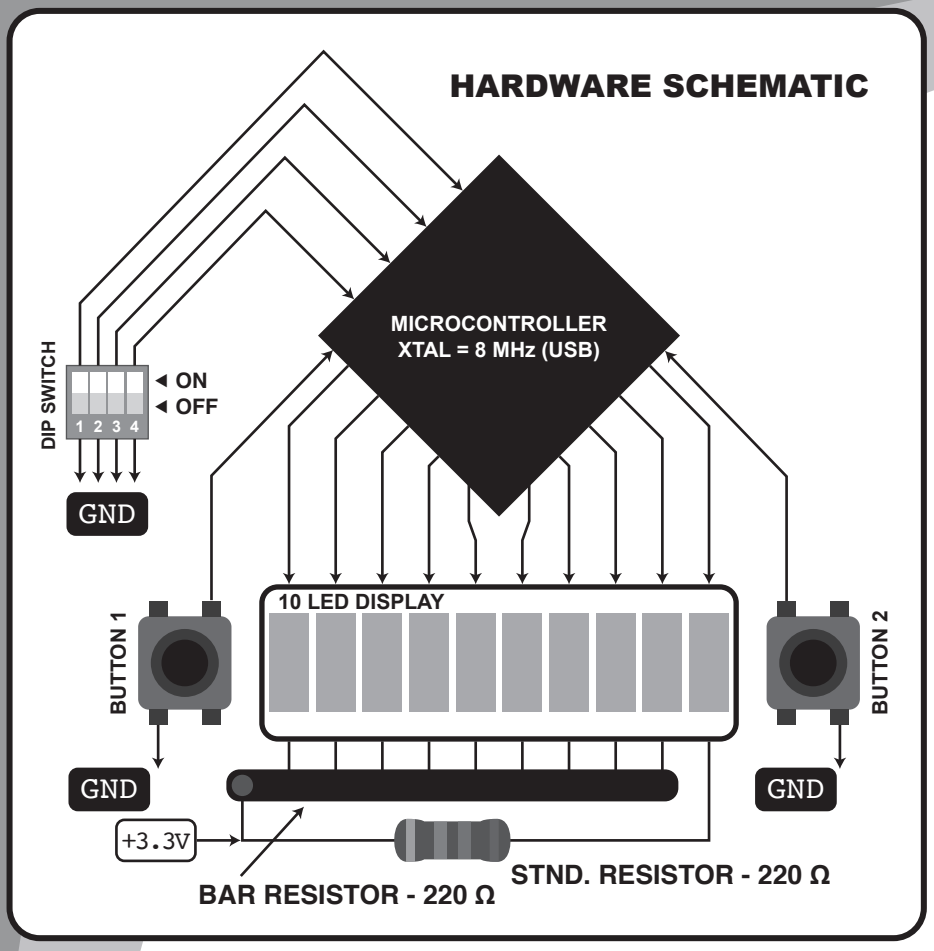
FINISH



Player 1 Wins Game!



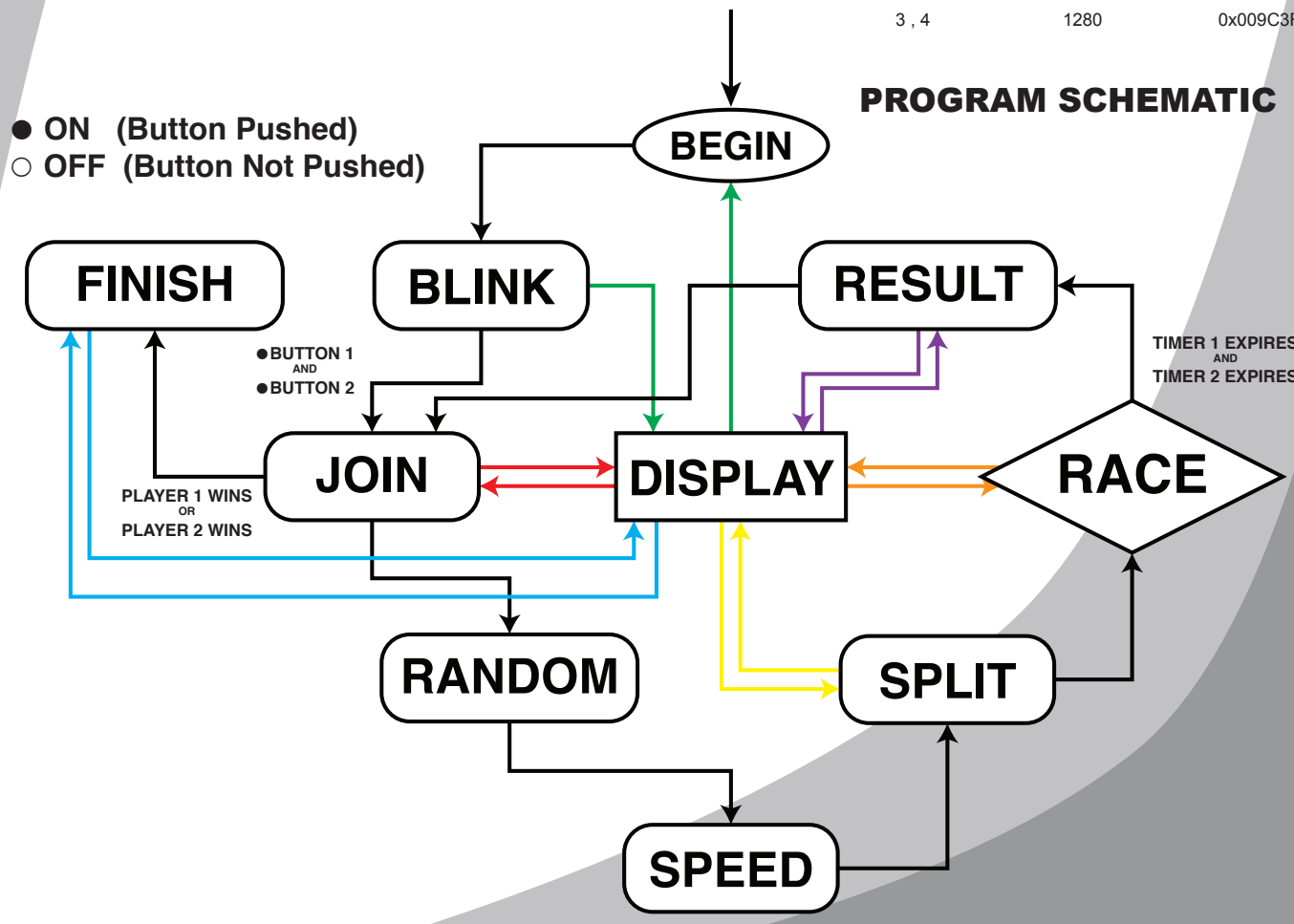
Player 2 Wins Game!



Timer Values

S_n (speed), d (no. of rounds)	T (delay time), ms	Timer, Initial (HEX)
0 , 0	320	0x00270FFF
1 , 0	240	0x001D4BFF
2 , 0	160	0x001387FF
3 , 0	80	0x0009C3FF
0 , 1	640	0x004E1FFF
1 , 1	480	0x003A97FF
2 , 1	320	0x00270FFF
3 , 1	160	0x001387FF
0 , 2	1280	0x009C3FFF
1 , 2	960	0x00752FFF
2 , 2	640	0x004E1FFF
3 , 2	320	0x00270FFF
0 , 3	2560	0x01387FFF
1 , 3	1920	0x00EA5FFF
2 , 3	1280	0x009C3FFF
3 , 3	640	0x004E1FFF
0 , 4	5120	0x0270FFFF
1 , 4	3840	0x01D4BFFF
2 , 4	2560	0x01387FFF
3 , 4	1280	0x009C3FFF

PROGRAM SCHEMATIC



Project SUMO

Overview

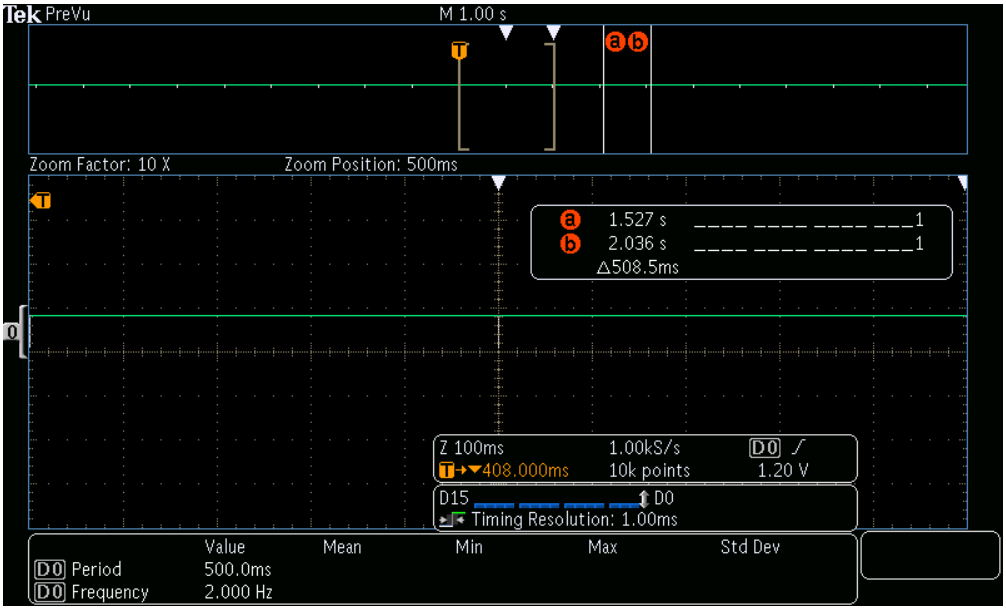
A two-player game using an LED bar graph and two push buttons to simulate a sumo wrestling match. Each player will control one of two sumo wrestlers and each will try to push the other out of the ring. As the match goes on, the wrestlers periodically shove each other apart, and the first to regain his balance is able to push the other a little closer to the edge of the ring.

During gameplay, the objective of the game is that each player needs to beat their own respective timer. When a player beats their timer and the other does not, the player that beat their timer moves his opponent back one space while the winning player moves forward one space. Play continues until one player manages to move the other player back to the edge of the LED display.

Requirements

1. The display shall consist of a 10-LED bar graph mounted horizontally.
2. There shall be 2 buttons, each in the proximity of a different end of the bar graph. Player 1 uses the button on the left and player 2 uses the button on the right.
3. There shall be a DIP switch to configure the speed of each player. The speed Sn for player n shall be interpreted as a 2-bit binary number, one switch per bit.
4. The buttons shall be sampled at least every 5 ms (milliseconds).
5. After the system is reset, the two center LEDs of the bar graph shall flash at a rate of 2 Hz. This rate will be controlled using a timer. The LED on the left represents player 1 and the LED on the right represents player 2.
6. Each player must press their button to indicate their readiness to play. Once a player presses their button, their LED shall be lit solidly.
7. At some random time at least 1 second but no more than 2 seconds after (a) both players indicate their readiness to play or (b) a move concludes that does not end the game, the leftmost lit LED shall move one spot to the left and the rightmost lit LED shall move one spot to the right. This event starts the move.
8. After the move starts, each player races to press their button. As soon as a button is pressed, the corresponding players lit LED moves back to its prior position and a timer is started.
9. If the timer in (8) expires before the opponent presses their button (and moves their lit LED), the quicker players lit LED shall move again and be adjacent to their opponent's lit LED. Otherwise, the move is a draw.
10. If the result of this move is that the two lit LEDs are on the leftmost or rightmost side of the bar graph, the game is over and the 2 lit LEDs shall ash at a rate of 2 Hz until the system is reset.
11. The delay time in (8) shall be based on the players speed, Sn, and the number of contiguous drawn moves, d. If player n is the first to press their button, the delay in milliseconds shall be $2^{-\min(d,4)}$ (320 – 80Sn).

Logic Analyzer Analysis



The LEDs blink at 2 Hz in the BEGIN and FINISH states which meets the timing specifications of the software requirements.

Program Code

```
       THUMB
       AREA    |.text|, CODE, READONLY, ALIGN=2
       EXPORT  Start
        ; Program Written by Joel Meine

Start

; MDS = Microcontroller Data Sheet (Texas Instruments Tiva TM4C123GH6PM)

;; Clock Configuration
  ; 1. Setup Clock
  LDR R1,=0x400FE000                    ; System Control Base Address : SYSCTL (pg. 235 - MDS)
  ;LDR R0,=0x078E3B80                   ; RCC:31.12 = Defaults
                                        ; BYPASS (RCC:11) = 1 _ PLL Bypass is Disabled.
                                        ; XTAL (RCC:10.6) = 0x0E _ 8 MHz (USB)
                                        ; OSCSRC (RCC:5.4) = 0x0 _ MOSC - Main Oscillator
                                        ; RCC:3.1 = Defaults
                                        ; MOSCDIS (RCC:0) = 0 _ Main Oscillator is Enabled.
  ;STR R0,[R1,#0x060]                   ; Run-Mode Clock Configuration : RCC (pg. 252 - MDS)
  ; 2. Enable Clocks
  MOV R0,#0x3                           ; Enable Clock on Timer 0, Timer 1 (TIMER0:16=1, TIMER1:17=1) (0x3 = 0b0011)
  STR R0,[R1,#0x106]                    ; Clock Base Address, GPTM : RCGC1 (pg. 458 - MDS)
  MOV R0,#0x15                          ; Enable Clock on PE, PC, PA (GPIOE:5=1, GPIOA:0=1) (0x15=0b0001.0101)
  STR R0,[R1,#0x108]                    ; Clock Base Address, GPIO : RCGC2 (pg. 462 - MDS)

;; Ports Configuration
; >> GPIO Port A (PA)                   ; 1 x 10 LED Bar Graph (LED:5.0)
  LDR R2,=0x40004000                    ; Port Base Address (pg. 656 - MDS)
  ; 1. Unlock Pins
  LDR R0,=0x4C4F434B                    ; Unlock Code (pg. 681 - MDS)
  STR R0,[R2,#0x520]                    ; GPIO Lock : GPIOLOCK (pg. 681 - MDS)
```

```
  MOV R0,#0xFC                          ; Unlock Pins - PA7, PA6, PA5, PA4, PA3, PA2 (0xFC=0b1111.1100)
  STR R0,[R2,#0x524]                    ; GPIO Commit : GPIOCR (pg. 682 - MDS)
  ; 2. Set Pins as Input or Output
  MOV R0,#0xFC                  ; Enable as Output - PA7, PA6, PA5, PA4, PA3, PA2 (LED:5.0) (0xFC=0b1111.1100)
  STR R0,[R2,#0x400]                    ; GPIO Direction : GPIODIR (pg. 660 - MDS)
  ; 3. Set for Open Drain (Outputs) and/or Pull-Up (Inputs)
  MOV R0,#0xFC                  ; Enable Open-Drain - PA7, PA6, PA5, PA4, PA3, PA2 (LED:5.0) (0xFC=0b1111.1100)
  STR R0,[R2,#0x50C]                    ; GPIO Open Drain Select : GPIODR (pg. 673 - MDS)
  ; 4. Enable Pins
  MOV R0,#0xFC                          ; Enable Pins - PA7, PA6, PA5, PA4, PA3, PA2 (0xFC=0b1111.1100)
  STR R0,[R2,#0x51C]                    ; GPIO Digital Enable : GPIODEN (pg. 679 - MDS)
  ; 5. Set Alias Addresses
                                        ; Port Alias Base Address (PaA)
                                        ; PaA = 0x42000000 + 32*0x43FC = 0x42087F80
                                        ; PA7 = PaA + 4*7 = 0x42087F9C (0x1C)
                                        ; PA6 = PaA + 4*6 = 0x42087F98 (0x18)
                                        ; PA5 = PaA + 4*5 = 0x42087F94 (0x14)
                                        ; PA4 = PaA + 4*4 = 0x42087F90 (0x10)
                                        ; PA3 = PaA + 4*3 = 0x42087F8C (0x0C)
                                        ; PA2 = PaA + 4*2 = 0x42087F88 (0x08)
                                        ; PA1 = PaA + 4*1 = 0x42087F84 (0x04)
                                        ; PA0 = PaA + 4*0 = 0x42087F80 (0x00)

; >> GPIO Port C (PC)                   ; 1 x DIP Switch (DIP:4.1)
  LDR R3,=0x40006000                    ; Port Base Address (pg. 656 - MDS)
  ; 1. Unlock Pins
  LDR R0,=0x4C4F434B                    ; Unlock Code (pg. 681 - MDS)
  STR R0,[R3,#0x520]                    ; GPIO Lock : GPIOLOCK (pg. 681 - MDS)
  MOV R0,#0xF0                          ; Unlock Pins - PC7, PC6, PC5, PC4 (0xF0=0b1111.0000)
  STR R0,[R3,#0x524]                    ; GPIO Commit : GPIOCR (pg. 682 - MDS)
  ; 2. Set Pins as Input or Output
  MOV R0,#0x0F                          ; Enable as Input - PC7, PC6, PC5, PC4 (DIP:4.1) (0x0F=0b0000.1111)
  STR R0,[R3,#0x400]                    ; GPIO Direction : GPIODIR (pg. 660 - MDS)
  ; 3. Set for Open Drain (Outputs) and/or Pull-Up (Inputs)
  MOV R0,#0xF0                          ; Enable Pull-Up - PC7, PC6, PC5, PC4 (DIP:4.1) (0xF0=0b1111.0000)
  STR R0,[R3,#0x510]                    ; GPIO Pull-Up Select : GPIOPUR (pg. 674 - MDS)
  ; 4. Enable Pins
  MOV R0,#0xF0                          ; Enable Pins - PC7, PC6, PC5, PC4 (0xF0=0b1111.0000)
  STR R0,[R3,#0x51C]                    ; GPIO Digital Enable : GPIODEN (pg. 679 - MDS)
  ; 5. Set Alias Addresses
                                        ; Port Alias Base Address (PaC)
                                        ; PaC = 0x42000000 + 32*0x63FC = 0x420C7F80
                                        ; PC7 = PaC + 4*7 = 0x420C7F9C (0x1C)
                                        ; PC6 = PaC + 4*6 = 0x420C7F98 (0x18)
                                        ; PC5 = PaC + 4*5 = 0x420C7F94 (0x14)
                                        ; PC4 = PaC + 4*4 = 0x420C7F90 (0x10)
                                        ; PC3 = PaC + 4*3 = 0x420C7F8C (0x0C)
                                        ; PC2 = PaC + 4*2 = 0x420C7F88 (0x08)
                                        ; PC1 = PaC + 4*1 = 0x420C7F84 (0x04)
                                        ; PC0 = PaC + 4*0 = 0x420C7F80 (0x00)

; >> GPIO Port E (PE)                   ; 1 x 10 LED Bar Graph (LED:9.6), 2 x Push Buttons (BUTTON1 & BUTTON2)
  LDR R4,=0x40024000                    ; Port Base Address (pg. 656 - MDS)
  ; 1. Unlock Pins
  LDR R0,=0x4C4F434B                    ; Unlock Code (pg. 681 - MDS)
  STR R0,[R4,#0x520]                    ; GPIO Lock : GPIOLOCK (pg. 681 - MDS)
  MOV R0,#0x3F                          ; Unlock Pins - PE5, PE4, PE3, PE2, PE1, PE0 (0x3F=0b0011.1111)
  STR R0,[R4,#0x524]                    ; GPIO Commit : GPIOCR (pg. 682 - MDS)
  ; 2. Set Pins as Input or Output
  MOV R0,#0xCF                          ; Enable as Output - PE3, PE2, PE1, PE0 (LED:9.6)
                                                        ; Enable as Input - PE5 (BUTTON2), PE4 (BUTTON1)
(0xCF=0b1100.1111)
  STR R0,[R4,#0x400]                    ; GPIO Direction : GPIODIR (pg. 660 - MDS)
  ; 3. Set for Open Drain (Outputs) and/or Pull-Up (Inputs)
  MOV R0,#0x0F                          ; Enable Open-Drain - PE3, PE2, PE1, PE0 (LED:9.6) (0x0F=0b0000.1111)
  STR R0,[R4,#0x50C]                    ; GPIO Open Drain Select : GPIODR (pg. 673 - MDS)
  MOV R0,#0x30                          ; Enable Pull-Up - PE5 (BUTTON2), PE4 (BUTTON1) (0x30=0b0011.0000)
```

```
  STR R0,[R4,#0x510]                    ; GPIO Pull-Up Select : GPIOPUR (pg. 674 - MDS)
  ; 4. Enable Pins
  MOV R0,#0x3F                          ; Enable Pins - PE5, PE4, PE3, PE2, PE1, PE0 (0x3F=0b0011.1111)
  STR R0,[R4,#0x51C]                    ; GPIO Digital Enable : GPIODEN (pg. 679 - MDS)
  ; 5. Set Alias Addresses
                                        ; Port Alias Base Address (PaE)
                                        ; PaE = 0x42000000 + 32*0x243FC = 0x42487F80
                                        ; PE7 = PaE + 4*7 = 0x42487F9C (0x1C)
                                        ; PE6 = PaE + 4*6 = 0x42487F98 (0x18)
                                        ; PE5 = PaE + 4*5 = 0x42487F94 (0x14)
                                        ; PE4 = PaE + 4*4 = 0x42487F90 (0x10)
                                        ; PE3 = PaE + 4*3 = 0x42487F8C (0x0C)
                                        ; PE2 = PaE + 4*2 = 0x42487F88 (0x08)
                                        ; PE1 = PaE + 4*1 = 0x42487F84 (0x04)
                                        ; PE0 = PaE + 4*0 = 0x42487F80 (0x00)

;; Timer Configuration
; >> SYSTICK (ST)                       ; Button Push Sampling (BPS), Flashing Light (FL), Random Time (RAND)
                                        ; BPS = 5 ms, FL = 500 ms, RAND = (1..2) s
  LDR R6,=0xE000E000                    ; Core Peripherals Base Address (pg. 132 - MDS)
  ; 1. Stop Timer (ENABLE=0)
  MOV R0,#0                             ; Stop Timer
  STR R0,[R6,#0x10]                     ; SysTick Control and Status Register : STCTRL (pg. 136 - MDS)
  ; 2. Set Initial Value (RELOAD=X)
  LDR R0,=0x00EA5F                      ; (RELOAD+1)*(1/CLK)=T2Z, CLK=Clock, T2Z=Time to Zero
                                        ; CLK=12 MHz, T2Z=5 ms
                                        ; RELOAD=#59999 (0x00EA5F)
  STR R0,[R6,#0x14]                     ; SysTick Reload Value Register : STRELOAD (pg. 138 - MDS)
  ; 3. Clear Current Value (write to CURRENT; clears count)
  MOV R0,#0                             ; Clear Count
  STR R0,[R6,#0x18]                     ; SysTick Current Value Register : STCURRENT (pg. 139 - MDS)
  LDR R0,[R6,#0x10]                     ; COUNT=0
  ; 4. Set Clock Source (core clock: CLK_SRC=1)
  ; 5. Enable/Disable Interrupts (INTEN=0)
  ; 6. Start Counting (ENABLE=1; sets CURRENT=RELOAD)
  MOV R0,#0x5                           ; CLK_SRC:2=1, INTEN:1=0, ENABLE:0=1, #0b0101 = #0x5
  STR R0,[R6,#0x10]                     ; SysTick Control and Status Register : STCTRL (pg. 136 - MDS)

; >> TIMER# (GPTM)
  ; 0. Enable Alt. Function             ; TIMER0.A is on PB6/PF0. TIMER1.A is on PB4/PF2.
  ;LDR R1,=0x40005000                   ; Port B (PB) Base Address (pg. 656 - MDS)
  ;MOV R0,#0x50                         ; Enable Pins and Alt. Function on Port. (0x40=0b01010000)
  ;STR R0,[R1,#0x420]                   ; GPIO Alternate Function Select : GPIOAFSEL (pg. 668 - MDS)
  ;STR R0,[R1,#0x51C]                   ; GPIO Digital Enable : GPIODEN (pg. 679 - MDS)
  ;LDR R1,=0x40025000                   ; Port F (PF) Base Address (pg. 656 - MDS)
  ;MOV R0,#0x05                         ; Enable Pins and Alt. Function on Port. (0x05=0b00000101)
  ;STR R0,[R1,#0x420]                   ; GPIO Alternate Function Select : GPIOAFSEL (pg. 668 - MDS)
  ;STR R0,[R1,#0x51C]                   ; GPIO Digital Enable : GPIODEN (pg. 679 - MDS)

; >> TIMER0 (TM0)                       ; Race Timer, Player 1 (RT1)
  LDR R7,=0x40030000                    ; Timer 0 Base Address (pg. 722 - MDS)
  ; 1. Stop Timer
  MOV R0,#0                             ; Disable Timer 0A (TAEN:0=0b0)
  STR R0,[R7,#0xC]                      ; GPTM Control : GPTMCTL (pg. 734 - MDS)
  ; 2. Select 32-/16-bit Timer
  MOV R0,#0x0                           ; 32-bit Timer (GPTMCFG:2.0=0b0000)
  STR R0,[R7,#0x0]                      ; GPTM Configuration : GPTMCFG (pg. 724 - MDS)
  ; 3. Select Timer Mode
  MOV R0,#0x5                           ; One-Shot Timer Mode (TAMR:1.0=0b01)
                                        ; Edge-Time Mode (TACMR:2=0b1)
  STR R0,[R7,#0x4]                      ; GPTM Timer A Mode : GPTMTAMR (pg. 726 - MDS)
  ; 4. Set Initial Value
  LDR R0,=0x00270FFF                    ; (INITIAL+1)*(1/CLK)=T2Z, CLK=Clock, T2Z=Time to Zero
                                        ; CLK=8 MHz, T2Z -> See "Timer Values"
  STR R0,[R7,#0x28]                     ; GPTM Timer A Interval Load : GPTMTAILR (pg. 753 - MDS)
  ; 5. Enable Timer & Start Counting
  ;MOV R0,#1                            ; Enable Timer 0A (TAEN:0=0b1)
```

```
  ;STR R0,[R7,#0xC]                     ; GPTM Control : GPTMCTL (pg. 734 - MDS)

; >> TIMER1 (TM1)                      ; Race Timer, Player 2 (RT2)
  LDR R8,=0x40031000                    ; Timer 1 Base Address (pg. 722 - MDS)
  ; 1. Stop Timer
  MOV R0,#0                             ; Disable Timer 1A (TAEN:0=0b0)
  STR R0,[R8,#0xC]                      ; GPTM Control : GPTMCTL (pg. 734 - MDS)
  ; 2. Select 32-/16-bit Timer
  MOV R0,#0x0                           ; 32-bit Timer (GPTMCFG:2.0=0b0000)
  STR R0,[R8,#0x0]                      ; GPTM Configuration : GPTMCFG (pg. 724 - MDS)
  ; 3. Select Timer Mode
  MOV R0,#0x5                           ; One-Shot Timer Mode (TAMR:1.0=0b01)
                                        ; Edge-Time Mode (TACMR:2=0b1)
  STR R0,[R8,#0x4]                      ; GPTM Timer A Mode : GPTMTAMR (pg. 726 - MDS)
  ; 4. Set Initial Value
  LDR R0,=0x00270FFF                    ; (INITIAL+1)*(1/CLK)=T2Z, CLK=Core Clock, T2Z=Time to Zero
                                        ; CLK=8 MHz, T2Z -> See "Timer Values"
  STR R0,[R8,#0x28]                     ; GPTM Timer A Interval Load : GPTMTAILR (pg. 753 - MDS)
  ; 5. Enable Timer & Start Counting
  ;MOV R0,#1                            ; Enable Timer 1A (TAEN:0=0b1)
  ;STR R0,[R8,#0xC]                     ; GPTM Control : GPTMCTL (pg. 734 - MDS)

  LDR R5,=0x20000000                    ; Memory (RAM) Base Address
                                        ; 0x100 = P1 Capture (RAND) = (0..100)
                                        ; 0x104 = P2 Capture (RAND) = (0..100)
                                        ; 0x108 = P1 Result (WIN=0 LOSE=1)
                                        ; 0x10C = P2 Result (WIN=0 LOSE=1)
                                        ; 0x110 = P1 Pushes = (0..2)
                                        ; 0x114 = P2 Pushes = (0..2)
                                        ; 0x118 = Count

; R0 - Empty
; R1 - Clock -> Empty
; R2 - Port A (PA)
; R3 - Port C (PC)
; R4 - Port E (PE)
; R5 - Memory (RAM)
; R6 - System Timer (BPS, FL, RAND)
; R7 - Timer 0 (RT1)
; R8 - Timer 1 (RT2)
; R9 - POSITION
; R10 - MOVES
; R11 - Player 1 Position
; R12 - Player 2 Position

BEGIN           ; START GAME
  MOV R9,#0x000                         ; Turn off LED display.
  BL.W DISPLAY                          ; Update LED Display.
  MOV R0,#0                             ; Start counter.
  STR R0,[R5,#0x118]                    ; Store count.
  STR R0,[R5,#0x110]                    ; Store number of player 1 button pushes.
  STR R0,[R5,#0x114]                    ; Store number of player 2 button pushes.
Breturn LDR R0,[R5,#0x118]              ; Load count.
  CMP R0,#201                           ; Does counter need to be reset?
  BNE Btimer                            ; If no, proceed to start timer (ST).
  MOV R0,#0                             ; If yes, then reset the counter.
  STR R0,[R5,#0x118]                    ; Store count.
Btimer MOV R0,#0x5                      ; Start timer (ST).
  STR R0,[R6,#0x10]                     ; ...
Bwait MOV R1,#1                         ; Timer Expired Flag
  LDR R0,[R6,#0x10]                     ; Check state of timer (ST).
  CMP R1,R0,LSR #16                     ; Has timer (ST) countdown reached zero? (COUNT:16=1)
  BNE Bwait                             ; If no, keep checking if timer (ST) has reached zero; i.e. 5 ms.
  MOV R0,#0                             ; Stop timer (ST).
  STR R0,[R6,#0x10]                     ; ...
  LDR R0,[R5,#0x118]                    ; Load count.
  ADD R0,#1                             ; Increment counter.
```

```
  STR R0,[R5,#0x118]                    ; Store count.
  LDR R0,[R4,#0x3FC]                    ; Check state of BUTTON1 (PE4) and BUTTON2 (PE5); ON=0 OFF=1.
  LSR R0,#4                             ; Isolate bits of BUTTON1 (PE4) and BUTTON2 (PE5).
                                                ; #0xF = Neither button is pushed.
                                                ; #0xE = Only player 1 button is pushed.
                                                ; #0xD = Only player 2 button is pushed.
                                                ; #0xC = Both buttons are pushed.
  CMP R0,#0xF                   ; Has neither BUTTON1 (PE4) or BUTTON2 (PE5) been pushed?
  BEQ.W BLINK                           ; Blink lights of both players.
  CMP R0,#0xE                   ; Is BUTTON1 (PE4) pushed?
  BEQ record1                           ; If yes, record counter value when BUTTON1 (PE4) was pushed.
  BNE skip1                             ; If no, check if player 2 pushed their button.
record1 LDR R1,[R5,#0x118]              ; Load count.
  STR R1,[R5,#0x100]            ; Store counter value of BUTTON1 (PE4).
  MOV R1,#1                             ; Record that BUTTON1 (PE4) was pushed.
  STR R1,[R5,#0x110]                    ; ...
  B.W BLINK                             ; Blink only light of player 2.
skip1 CMP R0,#0xD               ; Is BUTTON2 (PE5) pushed?
  BEQ record2                           ; If yes, record counter value when BUTTON2 (PE5) was pushed.
  BNE skip2                             ; If no, check if both players have pushed their button.
record2 LDR R1,[R5,#0x118]              ; Load count.
  STR R1,[R5,#0x104]                    ; Store counter value of BUTTON2 (PE5).
  MOV R1,#2                             ; Record that BUTTON2 (PE5) was pushed.
  STR R1,[R5,#0x114]                    ; ...
  B.W BLINK                             ; Blink only light of player 1.
skip2 CMP R0,#0xC               ; Have both BUTTON1 (PE4) and BUTTON2 (PE5) been pushed?
  BEQ record3                           ; If yes, record counter value.
  BNE Breturn                           ; If no, then check again for button pushes.
record3 LDR R1,[R5,#0x118]              ; Load count.
  STR R1,[R5,#0x100]                    ; Store counter value of BUTTON1 (PE4).
  STR R1,[R5,#0x104]                    ; Store counter value of BUTTON2 (PE5).
  MOV R1,#1                             ; Record that BUTTON1 (PE4) was pushed.
  STR R1,[R5,#0x110]                    ; ...
  MOV R1,#2                             ; Record that BUTTON2 (PE5) was pushed.
  STR R1,[R5,#0x114]                    ; ...
  B.W BLINK                             ; ...

DISPLAY         ; Updates LED:9.0 with present player positions.
  PUSH {LR}
  MVN R9,R9                             ; Convert position to active-low.
  LSL R0,R9,#2                          ; Shift Bits for Writing Only to PA:7-2 -- #2 means skipping PA:1-0.
  STR R0,[R2,#0x3FC]                    ; ...
  LSR R0,R0,#8                          ; Shift Bits for Writing Only to PE:3-0 -- #8 means storing on PE:3-0.
  STR R0,[R4,#0x3FC]                    ; ...
  MVN R9,R9                             ; Convert position to active-high.
  POP {LR}
  BX LR

BLINK           ; Blinks the light of player(s).
  LDR R0,[R5,#0x110]                    ; Check if player 1 pushed their button.
  LDR R1,[R5,#0x114]                    ; Check if player 2 pushed their button.
  ADD R0,R1                             ; ...
                                                ; #0 = Neither button was pushed.
                                                ; #1 = Only player 1 button was pushed.
                                                ; #2 = Only player 2 button was pushed.
                                                ; #3 = Both buttons were pushed.
  CMP R0,#0                     ; Did neither player push their button?
  BEQ blink0                            ; If yes, then blink both player lights.
  CMP R0,#1                     ; Did only player 1 push their button?
  BEQ blink1                            ; If yes, turn player 1 light solid.
  CMP R0,#2                     ; Did only player 2 push their button?
  BEQ blink2                            ; If yes, turn player 2 light solid.
  CMP R0,#3                     ; Did both players push their button?
  BEQ.W JOIN                            ; Both players have pushed their button.
blink0                  ; Blink lights of both players.
  LDR R1,[R5,#0x118]                    ; Load count.
  CMP R1,#100                           ; Has timer (ST) been repeated at least 100 times; i.e. 500 ms passed?
```

```
  MOVLS R9,#0x030                       ; If no, then lights of both players will be turned on.
  MOVHI R9,#0x000                       ; If yes, then lights of both players will be turned off.
  BL.W DISPLAY                          ; Update LED display.
  B.W Breturn                           ; Check again for button pushes.
blink1                  ; Turn player 1 light solid and blink light of player 2.
  LDR R1,[R5,#0x118]                    ; Load count.
  CMP R1,#100                           ; Has timer (ST) been repeated at least 100 times; i.e. 500 ms passed?
  MOVLS R9,#0x020                       ; If no, then display only light of player 1.
  MOVHI R9,#0x030                       ; If yes, then display lights of both players.
  BL.W DISPLAY                          ; Update LED display.
  B.W Breturn                           ; Check again for button pushes.
blink2                  ; Turn player 2 light solid and blink light of player 1.
  LDR R1,[R5,#0x118]                    ; Load count.
  CMP R1,#100                           ; Has timer (ST) been repeated at least 100 times; i.e. 500 ms passed?
  MOVLS R9,#0x010                       ; If no, then display only light of player 2.
  MOVHI R9,#0x030                       ; If yes, then display lights of both players.
  BL.W DISPLAY                          ; Update LED display.
  B.W Breturn                           ; Check again for button pushes.

  LTORG

JOIN            ; The light of each player is displayed next to each other.
  CMP R10,#0                            ; Did the game just begin?
  BNE Jnext                             ; If the game did not just begin...
  MOV R11,#0x020                        ; Update position of player 1.
  MOV R12,#0x010                        ; Update position of player 2.
  EOR R9,R11,R12                        ; Update POSITION.
  BL.W DISPLAY                          ; Update LED display.
  B Jskip                               ; ...
Jnext                                   ; ... then check if the game has ended.
  CMP R9,#0x003                         ; Did player 1 win the game?
  BEQ.W FINISH                                  ; If yes, it's the end of the game.
  CMP R9,#0x300                         ; Did player 2 win the game?
  BEQ.W FINISH                                  ; If yes, it's the end of the game.
Jskip B.W RANDOM

FINISH          ; END GAME
  ; Setup timer (ST) to start a countdown to on/off change of lights.
  MOV R1,#0                             ; Stop timer (ST).
  STR R1,[R6,#0x10]                     ; ...
  LDR R0,=0x5B8D7F                      ; (RELOAD+1)*(1/CC)=T2Z, CC=Core Clock, T2Z=Time to Zero
                                        ; CC=12 MHz, T2Z=500 ms
                                        ; RELOAD=#5999999 (0x5B8D7F)
  STR R0,[R6,#0x14]                     ; Set RELOAD for timer (ST).
FFrepeat STR R1,[R6,#0x18]              ; Clear timer (ST).
  MOV R0,#0x5                           ; Start timer (ST).
  STR R0,[R6,#0x10]                     ; ...
FFwait MOV R1,#1                        ; Timer Expired Flag
  LDR R0,[R6,#0x10]                     ; Check state of timer (ST).
  CMP R1,R0,LSR #16                     ; Has timer (ST) countdown reached zero? (COUNT:16=1)
  BNE FFwait                            ; If no, keep checking if timer (ST) has reached zero; i.e. 500 ms.
  MOV R0,#0                             ; Stop timer (ST).
  STR R0,[R6,#0x10]                     ; ...
  CMP R9,#0x003                 ; Did player 1 win the game? (POSITION = 0x003=0b0000000011)
  BEQ P1won                             ; If yes, turn on/off LED.1 and LED.0.
  CMP R9,#0x300                 ; Did player 2 win the game? (POSITION = 0x300=0b1100000000)
  BEQ P2won                             ; If yes, turn on/off LED.8 and LED.9.
P1won
  CMP R9,#0x000                         ; Are player lights off? (ON=1 OFF=0)
  ITT EQ                                ; ...
  MOVEQ R9,#0x003                       ; If yes, then turn player lights on.
  BEQ FFskip1                           ; ...
  CMP R9,#0x003                         ; Are player lights on? (ON=1 OFF=0)
  MOVEQ R9,#0x000                       ; If yes, then turn player lights off.
FFskip1 BL.W DISPLAY                    ; Update LED display.
  B FFrepeat
P2won
```

```
  CMP R9,#0x000                         ; Are player lights off? (ON=1 OFF=0)
  ITT EQ                                ; ...
  MOVEQ R9,#0x300                       ; If yes, then turn player lights on.
  BEQ FFskip2                           ; ...
  CMP R9,#0x300                         ; Are player lights on? (ON=1 OFF=0)
  MOVEQ R9,#0x000                       ; If yes, then turn player lights off.
FFskip2 BL.W DISPLAY                    ; Update LED display.
  B FFrepeat

RANDOM          ; A random time between 1 s and 2 s is selected.
  LDR R0,[R5,#0x100]                    ; Load counter value of BUTTON1 (PE4).
  LDR R1,[R5,#0x104]                    ; Load counter value of BUTTON2 (PE5).
  CMP R0,#99                            ; Does counter value of BUTTON1 (PE4) require adjustment?
  SUBHI R0,#100                         ; If yes, then make the adjustment.
  CMP R1,#99                            ; Does counter value of BUTTON2 (PE5) require adjustment?
  SUBHI R1,#100                         ; If yes, then make the adjustment.
  CMP R0,R1                             ; Which player has the lower counter value?
  BHI Rnext                             ; If player 1 does not have the lower counter value...
  MOV R0,R0                             ; Use the counter value of player 1.
  B Rskip                               ; ...
Rnext MOV R0,R1                         ; ... then use the counter value of player 2.
Rskip MOV R1,#10                        ; ...
  MUL R0,R1                             ; 10*count
  ADD R0,#1000                          ; 1000+10*count; i.e. Random Time (RAND), ms
  MOV R1,#12000                         ; Clock, SYSTICK (CLKS) = 12 MHz
  MUL R0,R1                             ; RAND*CLKS
  SUB R0,#2                             ; RAND*CLKS - 2
  LSR R0,#1                             ; (RAND*CLKS-2)/2; i.e. RELOAD
  ; Setup timer (ST) to start a countdown to the light split.
  MOV R1,#0                             ; Stop timer (ST).
  STR R1,[R6,#0x10]                     ; ...
  STR R0,[R6,#0x14]                     ; Set RELOAD for timer (ST).
  STR R1,[R6,#0x18]                     ; Clear timer (ST).
  LDR R1,[R6,#0x10]                     ; ...
  MOV R0,#0                             ; Start counter.
  STR R0,[R5,#0x118]                    ; Store count.
timerRAND MOV R0,#0x5                   ; Start timer (ST).
  STR R0,[R6,#0x10]                     ; ...
Rwait
  MOV R1,#1                             ; Timer Expired Flag
  LDR R0,[R6,#0x10]                     ; Check state of timer (ST).
  CMP R1,R0,LSR #16                     ; Has timer (ST) countdown reached zero? (COUNT:16=1)
  BNE Rwait                             ; If no, keep checking if timer (ST) has reached zero; i.e. 5 ms.
  MOV R0,#0                             ; Stop timer (ST).
  STR R0,[R6,#0x10]                     ; ...
  LDR R0,[R5,#0x118]                    ; Load count.
  ADD R0,#1                             ; Increment counter.
  STR R0,[R5,#0x118]                    ; Store count.
  LDR R0,[R5,#0x118]                    ; Load count.
  CMP R0,#2                             ; Has timer (ST) been repeated 2 times?
  BNE timerRAND                         ; If not, then start timer again.
  B.W SPEED

SPEED           ; Updates player delays by speed setting and number of MOVES.
  CMP R10,#4                            ; Is number of MOVES at least four?
  MOVHS R0,#16                                  ; If yes, scale moves to sixteen; i.e. 2^4.
  CMP R10,#3                            ; Is number of MOVES three?
  MOVEQ R0,#8                                   ; If yes, scale moves to eight; i.e. 2^3.
  CMP R10,#2                            ; Is number of MOVES two?
  MOVEQ R0,#4                                   ; If yes, scale moves to four; i.e. 2^2.
  CMP R10,#1                            ; Is number of MOVES one?
  MOVEQ R0,#2                                   ; If yes, scale moves to two; i.e. 2^1.
  CMP R10,#0                            ; Is number of MOVES zero?
  MOVEQ R0,#1                                   ; If yes, scale moves to one; i.e. 2^0.
  STR R0,[R5,#0x11C]                    ; Store scaled moves.
  ; Calculate INITIAL value for timer of player 1.
  LDR R0,[R3,#0x3FC]                    ; Check state of DIP:4.1 (PC:7.4); ON=1 OFF=0.
```

```
  LSR R0,#4                           ; Isolate bits of DIP:4.1 (PC:7.4).
  AND R0,#0x3                         ; Isolate bits of DIP:2.1 (PC:5.4).
                                      ; Player 1 Speed (S1) = DIP:1 + DIP:2
  MOV R1,#80                          ; ...
  MUL R0,R1                           ; 80*S1
  MOV R1,#320                         ; ...
  SUB R0,R1,R0                        ; 320-80*S1
  LDR R1,[R5,#0x11C]                  ; Load scaled moves.
  MUL R1,R1,R0                        ; (2^MOVES)*(320-80*S1); i.e. Player 1 Timer (P1T), ms
  MOV R0,#8000                        ; Clock, GPTM (CLKG) = 8 MHz
  MUL R0,R1                           ; P1T*CLKG
  SUB R0,#1                           ; P1T*CLKG - 1; i.e. INITIAL
  STR R0,[R7,#0x28]                   ; Set INITIAL value for timer (RT1).
  ; Calculate INITIAL value for timer of player 2.
  LDR R0,[R3,#0x3FC]                  ; Check state of DIP:4.1 (PC:7.4); ON=0 OFF=1.
  LSR R0,#6                           ; Isolate bits of DIP:4.3 (PC:7.6).
                                      ; Player 2 Speed (S2) = DIP:3 + DIP:4
  MOV R1,#80                          ; ...
  MUL R0,R1                           ; 80*S2
  MOV R1,#320                         ; ...
  SUB R0,R1,R0                        ; 320-80*S2
  LDR R1,[R5,#0x11C]                  ; Load scaled moves.
  MUL R1,R1,R0                        ; (2^MOVES)*(320-80*S2); i.e. Player 2 Timer (P2T), ms
  MOV R0,#8000                        ; Clock, GPTM (CLKG) = 8 MHz
  MUL R0,R1                           ; P2T*CLKG
  SUB R0,#1                           ; P2T*CLKG - 1; i.e. INITIAL
  STR R0,[R8,#0x28]                   ; Set INITIAL value for timer of player 2.
  B.W SPLIT

SPLIT           ; The light of each player is displayed separated from each other.
  CMP R10,#0                          ; Did the game just begin?
  BNE Snext                           ; If the game did not just begin...
  MOV R9,#0x048                       ; POSITION = 0x048=0b0001001000
  BL.W DISPLAY                        ; Update LED display.
  B Sskip                             ; ...
Snext                                 ; ... then move back players and update LED display.
  LSL R11,#1                          ; Move player 1 back one space.
  LSR R12,#1                          ; Move player 2 back one space.
  EOR R9,R11,R12                      ; Update POSITION.
  BL.W DISPLAY                        ; Update LED display.
Sskip B.W RACE                        ; ...

RACE            ; The two players race to push their button.
  LDR R0,=0x00EA5F                    ; Load RELOAD for 5 ms.
  STR R0,[R6,#0x14]                   ; Set RELOAD for timer (ST).
  MOV R0,#1                           ; Start counter to track player button pushes.
  STR R0,[R5,#0x110]                  ; Store number of player 1 button pushes.
  STR R0,[R5,#0x114]                  ; Store number of player 2 button pushes.
  MOV R0,#0                           ; Start counter.
  STR R0,[R5,#0x118]                  ; Store counter.
Freturn LDR R0,[R5,#0x118]            ; Load counter.
  CMP R0,#201                         ; Does counter need to be reset?
  BNE Ftimer                          ; If no, proceed to start timer (ST).
  MOV R0,#0                           ; If yes, then reset the counter.
  STR R0,[R5,#0x118]                  ; Store count.
Ftimer MOV R0,#0                      ; Clear count.
  STR R0,[R6,#0x18]                   ; Clear timer (ST).
  MOV R0,#0x5                         ; Start timer (ST).
  STR R0,[R6,#0x10]                   ; ...
Fwait MOV R1,#1                       ; Timer Expired Flag
  LDR R0,[R6,#0x10]                   ; Check state of timer (ST).
  CMP R1,R0,LSR #16                   ; Has timer (ST) countdown reached zero? (COUNT:16=1)
  BNE Fwait                           ; If no, keep checking if timer (ST) has reached zero; i.e. 5 ms.
  MOV R0,#0                           ; Stop timer (ST).
  STR R0,[R6,#0x10]                   ; ...
  LDR R0,[R5,#0x118]                  ; Load count.
  ADD R0,#1                           ; Increment counter.
```

```
  STR R0,[R5,#0x118]                    ; Store count.
  ; Check button pushes of players and record their results.
  LDR R0,[R4,#0x3FC]                    ; Check state of BUTTON1 (PE4) and BUTTON2 (PE5); ON=0 OFF=1.
  LSR R0,#4                             ; Isolate bits of BUTTON1 (PE4) and BUTTON2 (PE5).
                                        ; #0xF = Neither button is pushed.
                                        ; #0xE = Only player 1 button is pushed.
                                        ; #0xD = Only player 2 button is pushed.
                                        ; #0xC = Both buttons are pushed.
  CMP R0,#0xF                   ; Has neither BUTTON1 (PE4) or BUTTON2 (PE5) been pushed?
  BEQ Freturn                           ; If yes, check again for button pushes.
; Check button pushes of player 1 and record their result.
  CMP R0,#0xE                   ; Is BUTTON1 (PE4) pushed?
  BEQ timerP1                           ; If yes, check if this is first or second button push for player 1.
  BNE Fnext2                            ; If no, proceed to check button of other player.
timerP1 LDR R1,[R5,#0x110]              ; Load number of player 1 button pushes.
  CMP R1,#1                             ; Is this the first button push for player 1?
  BEQ timerRT1                          ; If yes, then start timer of player 1.
  CMP R1,#2                             ; Is this the second button push for player 1?
  BEQ Fresult1                          ; If yes, then check if player 1 beat their timer.
  BHI Fnext2                            ; If no, proceed to check button of other player.
timerRT1 MOV R0,#1                      ; ...
  STR R0,[R7,#0xC]                      ; Start timer (RT1).
  ADD R1,#1                             ; Player 1 has pushed their button one time.
  STR R1,[R5,#0x110]                    ; Store number of player 1 button pushes.
  B Fnext2                              ; Proceed to check button of other player.
Fresult1 ADD R1,#1                      ; Player 1 has pushed their button two times.
  STR R1,[R5,#0x110]                    ; Store number of player 1 button pushes.
  LDR R0,[R5,#0x118]                    ; Load count.
  STR R0,[R5,#0x100]                    ; Store counter value of BUTTON1 (PE4).
  LDR R0,[R7,#0x1C]             ; Timer Has not Timed Out=0 (WIN), Timer Has Timed Out=1 (LOSE) (pg. 747 - MDS)
  CMP R0,#1                             ; Did timer of player 1 expire?
  BEQ P1expire                          ; If yes, then move light of player 1 forward one space.
  BNE P1result                          ; If no, then record if player 1 beat their timer or not.
P1expire LSR R11,#1                     ; Move player 1 forward one space.
  EOR R9,R11,R12                        ; Update POSITION.
  BL.W DISPLAY                          ; Update LED display.
P1result STR R0,[R5,#0x108]             ; Store the result of player 1. (WIN=0 LOSE=1)
; Check button pushes of player 2 and record their result.
Fnext2 CMP R0,#0xD              ; Is BUTTON2 (PE5) pushed?
  BEQ timerP2                           ; If yes, then start timer of player 2.
  BNE Fnext3                            ; If no, check if both players pushed their button simultaneously.
timerP2 LDR R1,[R5,#0x114]              ; Load number of player 2 button pushes.
  CMP R1,#1                             ; Is this the first button push for player 2?
  BEQ timerRT2                          ; If yes, then start timer of player 2.
  CMP R1,#2                             ; Is this the second button push for player 2?
  BEQ Fresult2                          ; If yes, then check if player 2 beat their timer.
  BHI Rdone                             ; If no, check if both timers have expired.
timerRT2 MOV R0,#1                      ; ...
  STR R0,[R8,#0xC]                      ; Start timer (RT2).
  ADD R1,#1                             ; Player 2 has pushed their button one time to start timer (RT2).
  STR R1,[R5,#0x114]                    ; Store number of player 2 button pushes.
  B Rdone                               ; Check if both timers have expired.
Fresult2 ADD R1,#1                      ; Player 2 has pushed their button two times.
  STR R1,[R5,#0x114]                    ; Store number of player 2 button pushes.
  LDR R0,[R5,#0x118]                    ; Load count.
  STR R0,[R5,#0x104]                    ; Store counter value of BUTTON2 (PE5).
  LDR R0,[R8,#0x1C]             ; Timer Has not Timed Out=0 (WIN), Timer Has Timed Out=1 (LOSE) (pg. 747 - MDS)
  CMP R0,#1                                             ; Did timer of player 2 expire?
  BEQ P2expire                          ; If yes, then move light of player 2 forward one space.
  BNE P2result                          ; If no, then record if player 2 beat their timer or not.
P2expire LSL R12,#1                     ; Move player 2 forward one space.
  EOR R9,R11,R12                        ; Update POSITION.
  BL.W DISPLAY                          ; Update LED display.
P2result STR R0,[R5,#0x10C]             ; Store the result of player 2. (WIN=0 LOSE=1)
; Check if both players pushed their button simultaneously.
Fnext3 CMP R0,#0xC              ; Have both BUTTON1 (PE4) and BUTTON2 (PE5) been pushed?
  BEQ timerP12                          ; If yes, then start timer of player 1 and player 2.
```

```
  BNE Rdone                             ; If no, check if both timers have expired.
timerP12 LDR R1,[R5,#0x110]             ; Load number of player 1 button pushes.
  CMP R1,#1                             ; Is this the first button push for player 1?
  BEQ timerRT11                         ; If yes, then start timer of player 1.
  CMP R1,#2                             ; Is this the second button push for player 1?
  BEQ Fresult11                         ; If yes, then check if player 1 beat their timer.
  BHI Rdone                             ; If no, check if both timers have expired.
timerP21 LDR R1,[R5,#0x114]             ; Load number of player 2 button pushes.
  CMP R1,#1                             ; Is this the first button push for player 2?
  BEQ timerRT22                         ; If yes, then start timer of player 2.
  CMP R1,#2                             ; Is this the second button push for player 2?
  BEQ Fresult22                         ; If yes, then check if player 2 beat their timer.
  BHI Rdone                             ; If no, check if both timers have expired.
timerRT11 MOV R0,#1                     ; ...
  STR R0,[R7,#0xC]                      ; Start timer (RT1).
  ADD R1,#1                             ; Player 1 has pushed their button one time to start timer (RT1).
  STR R1,[R5,#0x110]                    ; Store number of player 1 button pushes.
  B timerP21                            ; ...
timerRT22 MOV R0,#1                     ; ...
  STR R0,[R8,#0xC]                      ; Start timer (RT2).
  ADD R1,#1                             ; Player 2 has pushed their button one time to start timer (RT2).
  STR R1,[R5,#0x114]                    ; Store number of player 2 button pushes.
  B Rdone                               ; Check if both timers have expired.
Fresult11 ADD R1,#1                     ; Player 1 has pushed their button two times.
  STR R1,[R5,#0x110]                    ; Store number of player 1 button pushes.
  LDR R0,[R5,#0x118]                    ; Load count.
  STR R0,[R5,#0x100]                    ; Store counter value of BUTTON1 (PE4).
  LDR R0,[R7,#0x1C]             ; Timer Has not Timed Out=0 (WIN), Timer Has Timed Out=1 (LOSE) (pg. 747 - MDS)
  CMP R0,#1                             ; Did timer of player 1 expire?
  BEQ P11expire                         ; If yes, then move light of player 1 forward one space.
  BNE P11result                         ; If no, then record if player 1 beat their timer or not.
P11expire LSR R11,#1                    ; Move player 1 forward one space.
  EOR R9,R11,R12                        ; Update POSITION.
  BL.W DISPLAY                          ; Update LED display.
P11result STR R0,[R5,#0x108]            ; Store the result of player 1. (WIN=0 LOSE=1)
  B timerP21                            ; ...
Fresult22 ADD R1,#1                     ; Player 2 has pushed their button two times.
  STR R1,[R5,#0x114]                    ; Store number of player 2 button pushes.
  LDR R0,[R5,#0x118]                    ; Load count.
  STR R0,[R5,#0x104]                    ; Store counter value of BUTTON2 (PE5).
  LDR R0,[R8,#0x1C]             ; Timer Has not Timed Out=0 (WIN), Timer Has Timed Out=1 (LOSE) (pg. 747 - MDS)
  CMP R0,#1                             ; Did timer of player 2 expire?
  BEQ P22expire                         ; If yes, then move light of player 2 forward one space.
  BNE P22result                         ; If no, then record if player 12beat their timer or not.
P22expire LSL R12,#1                    ; Move player 2 forward one space.
  EOR R9,R11,R12                        ; Update POSITION.
  BL.W DISPLAY                          ; Update LED display.
P22result STR R0,[R5,#0x10C]            ; Store the result of player 2. (WIN=0 LOSE=1)
; Check if timer of both players has expired.
Rdone LDR R0,[R7,#0x1C]                 ; Check if timer of player 1 expired.
  LDR R1,[R8,#0x1C]                     ; Check if timer of player 2 expired.
  AND R0,R0,R1                          ; ...
  CMP R0,#1                     ; Have the timers expired for both players?
  BEQ.W RESULT                          ; If yes, then determine the winner of the race.
  BNE.W Freturn                         ; If no, then check again for button pushes.

RESULT          ; The result of the race is determined.
  LDR R0,[R5,#0x108]                    ; Load race result of player 1.
  LSL R0,#1                             ; Shift bit from LSB to MSB.
  LDR R1,[R5,#0x10C]                    ; Load race result of player 2.
  ADD R0,R1                             ; Set the four possible outcomes of the race.
                                                ; #0=(P1=WIN & P2=WIN) #1=(P1=WIN & P2=LOSE)
                                                ; #2=(P1=LOSE & P2=WIN) #3=(P1=LOSE & P2=LOSE)
  CMP R0,#0                     ; Did both players beat their timers?
  BEQ.W JOIN                            ; If yes, the race is a draw and another race is started.
  CMP R0,#3                     ; Did both players not beat their timers?
  BEQ.W JOIN                            ; If yes, the race is a draw and another race is started.
```

```
  CMP R0,#1                     ; Did player 1 win the race?
  BEQ won1                              ; ...
  BNE RRnext                            ; If no, then check if player 2 won.
won1 LSR R11,#1                         ; If yes, then player 1 moves forward one space...
  LSR R12,#1                            ; ...and player 2 moves back one space.
  EOR R9,R11,R12                        ; Update POSITION.
  BL DISPLAY                            ; Update LED display.
  B.W JOIN                              ; Another race is started.
RRnext CMP R0,#2                ; Did player 2 win the race?
  BEQ won2                              ; ...
  BNE.W JOIN                            ; If no, then another race is started.
won2 LSL R12,#1                         ; If yes, then player 2 moves forward one space...
  LSL R11,#1                            ; ...and player 1 moves back one space.
  EOR R9,R11,R12                        ; Update POSITION.
  BL DISPLAY                            ; Update LED display.
  B.W JOIN                              ; Another race is started.

      ALIGN
      END
```