LFSR and ASG Pseudorandom Number Generators
Embedded System Design, Lab 5

Ben Lorenzetti

October 22, 2015

Contents

1 Objectives and Problem Descriptions **5**
1.1 8-Bit Linear Feedback Shift Register (LFSR) 5
1.2 Alternating Step Generator (ASG) 5

2 Procedure **5**
2.1 Reading Assignments 5
2.2 LFSRs and ASGs Theory 5
2.3 Implementation Design 5
2.4 Hello World Program (Delay Function) 5
2.5 Assembly Macros (Rotate Word Function) 5
2.6 XOR Propagation Function 5
2.7 8-Bit LFSR Design 5
2.8 ASG Design 5

3 Expected Results **5**
3.1 3-Bit LFSR Demonstration 5
3.2 8-Bit LFSR Demonstration 5
3.3 ASG Demonstration 5

4 Experiment and Design Revisions **5**
4.1 Addition of Macros 5
4.2 Use of Carry Flag between Functional Blocks 5

5 Observations **5**
5.1 Observations 5

6 Discussion **5**
6.1 Discussion of Results 5
6.2 Carry Flag and PIC Design 5

7 Exercises **5**

8 Implementation Code **7**
8.1 8-Bit LFSR 7
8.2 Alternating Step Generator 9

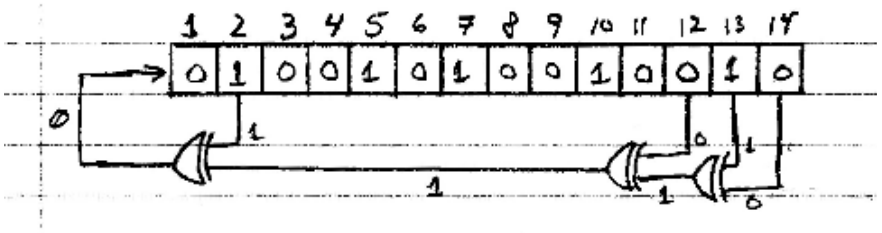

Figure 1: A 14-bit Fibonacci LFSR with feedback polynomial $F(x) = x^{14} + x^{13} + x^{12} + x^2 + 1$

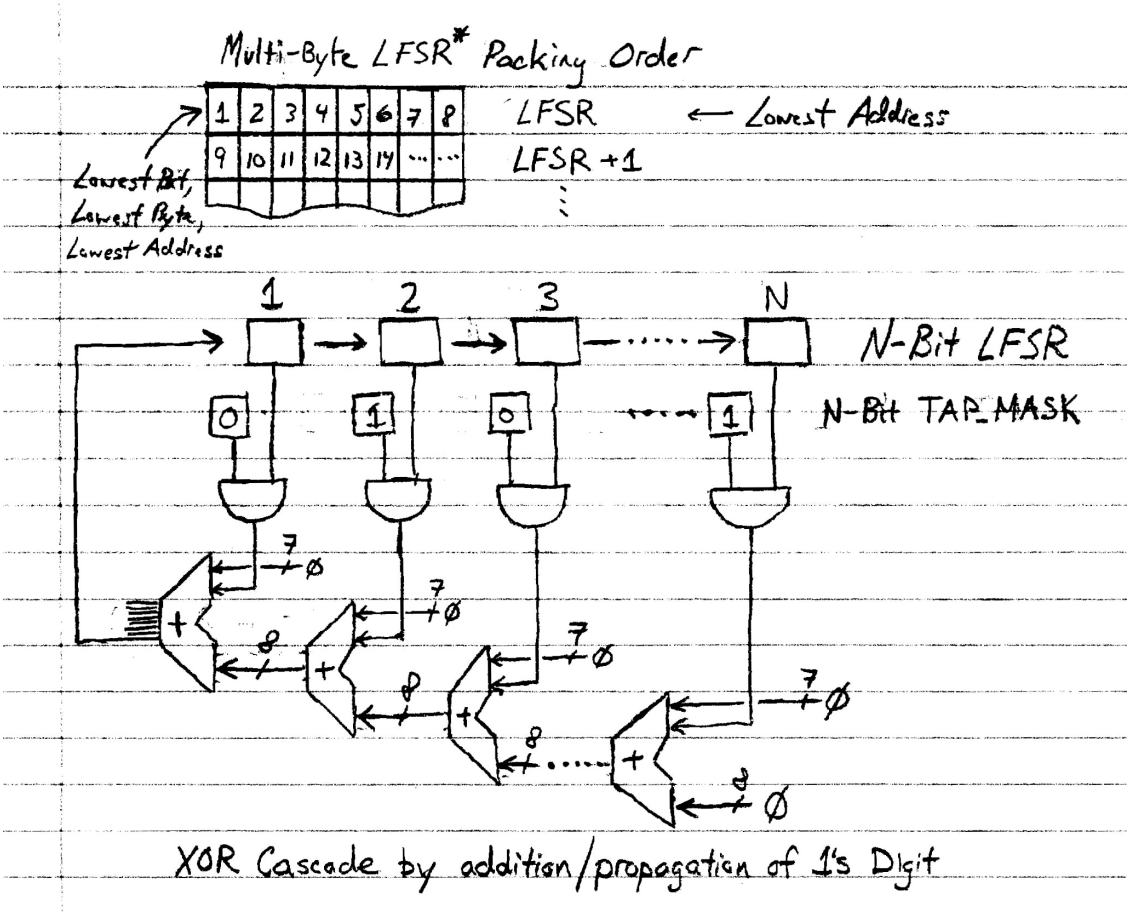

Figure 2: Equivalent Circuit Implementation of an Arbitrary Length Fibonacci LFSR

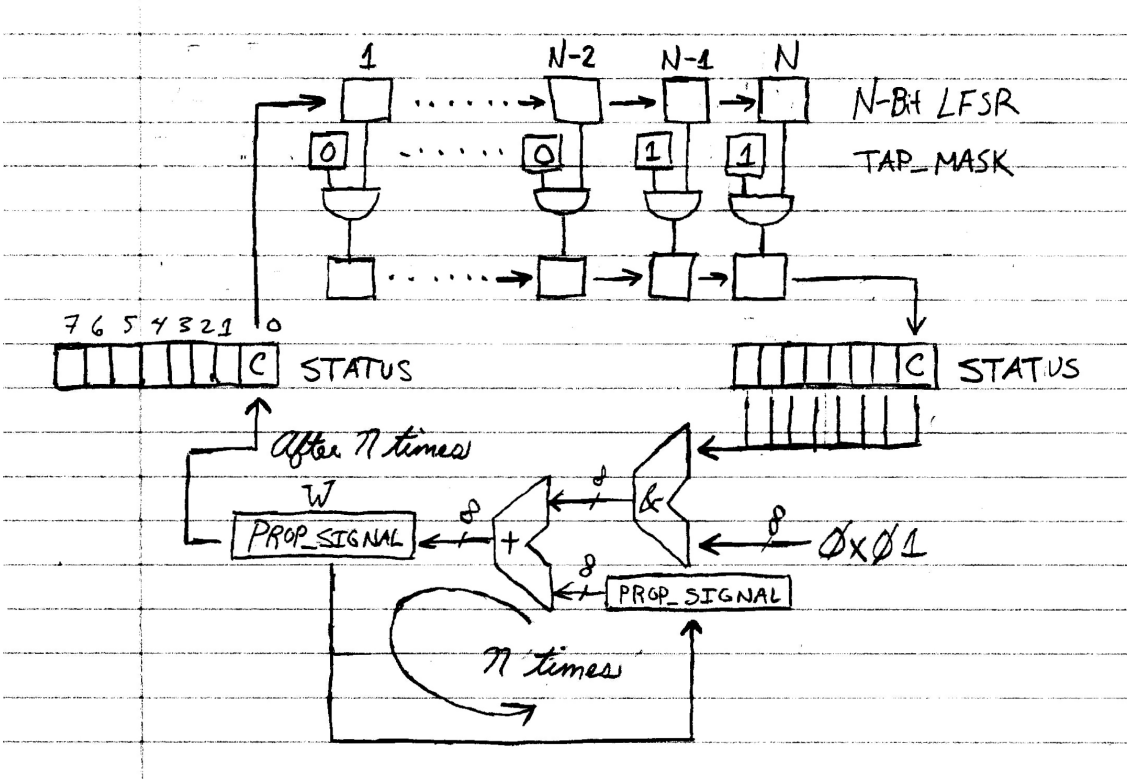

Figure 3: Implementation of XOR Propagation

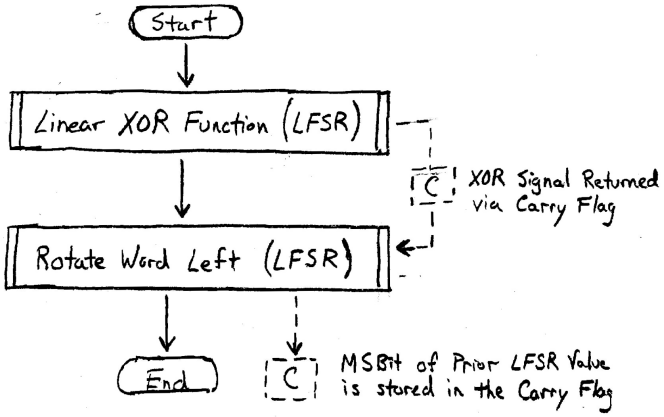

Figure 4: Cycle LFSR Function Implementation Flowchart

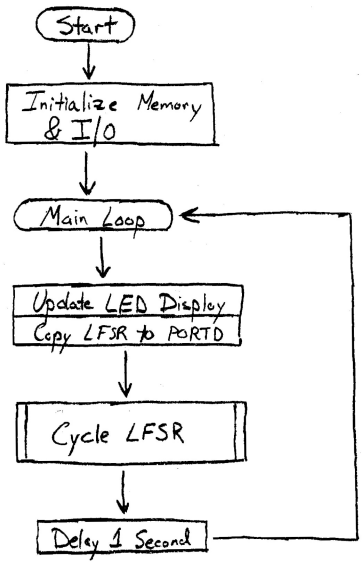

Figure 5: 8-Bit LFSR Implementation Flowchart

1 Objectives and Problem Descriptions

1.1 8-Bit Linear Feedback Shift Register (LFSR)

1.2 Alternating Step Generator (ASG)

2 Procedure

2.1 Reading Assignments

2.2 LFSRs and ASGs Theory

2.3 Implementation Design

2.4 Hello World Program (Delay Function)

2.5 Assembly Macros (Rotate Word Function)

2.6 XOR Propagation Function

2.7 8-Bit LFSR Design

2.8 ASG Design

3 Expected Results

3.1 3-Bit LFSR Demonstration

3.2 8-Bit LFSR Demonstration

3.3 ASG Demonstration

4 Experiment and Design Revisions

4.1 Addition of Macros

4.2 Use of Carry Flag between Functional Blocks

5 Observations

5.1 Observations

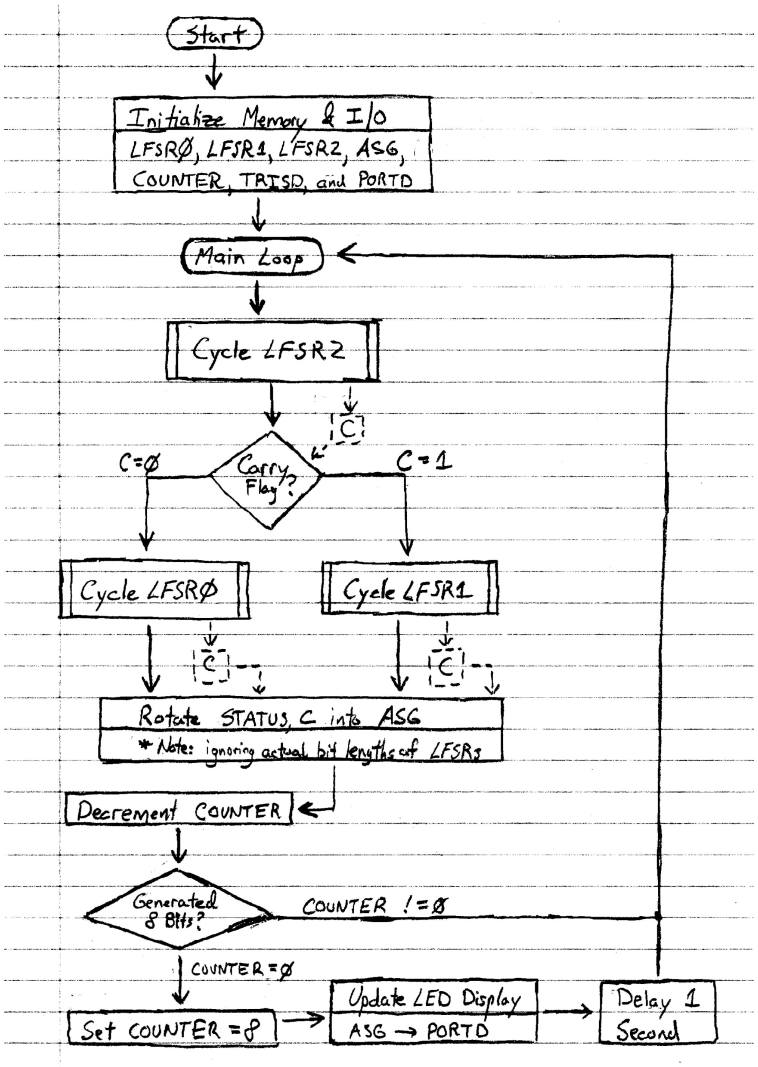

Figure 6: Alternating Step Generator Implementation

8 Implementation Code

8.1 8-Bit LFSR

```
; lfsr.asm
; 8-Bit Linear Feedback Shift Register on PIC Development Board
; Ben Lorenzetti
; Embedded Systems Design, Fall 2015

#include <p16f887.inc>
        __CONFIG        _CONFIG1, _LVP_OFF & _FCMEN_OFF & _IESO_OFF &
           _BOR_OFF & _CPD_OFF & _CP_OFF & _MCLRE_OFF & _PWRTE_ON &
           _WDT_OFF & _INTRC_OSC_NOCLKOUT
        __CONFIG        _CONFIG2, _WRT_OFF & _BOR21V

;----------LFSR Bit/Byte Sizes, Tap Locations, and Initial Values
   ----------;
#define LFSR_SIZE 1             ; LFSR byte size
#define LED_MASK        0x0F    ; 0x07 for 3 bit LFSR, 0x0F for 4-bit
   LFSR, 0xFF for 8-bit LFSR
#define LFSR1_TAP_MASK  0x0C    ; 0x06 for 3-bit LFSR, 0x0C for 4-bit
   LFSR, 0xB8 for 8-bit LFSR
#define INITIAL_VALUE   1

;-------------------------Organize Program Memory-------------------------;
Reset_Vector
        ORG 0
        GOTO Initialize

Interupt_Vector
        ORG 4

;---------------------Allocate Static Variables-------------------------;
        cblock  0x20
        OUTER_DELAY
        MIDDLE_DELAY
        INNER_DELAY
        REMAINDER_DELAY
        PROP_SIGNAL
        BIT_INDEX
        LOCAL_LFSR: LFSR_SIZE
        LFSR1: LFSR_SIZE
        endc

;-------- void rotate_word_left (word*, word_size); ------------------;
RLF_Word        MACRO   Word_Addr, Word_Size
        local i = 0
        while i < Word_Size
```

```
            RLF     (Word_Addr + i), 1
            i += 1
        endw
        ENDM

#define OUTER_MAX_PLUS_1 60
#define MIDDLE_MAX_PLUS_1 40
#define INNER_MAX_PLUS_1 40
#define REMAINDER 1
;---------------------- void Pause_1_Second () ------------------------;
Pause_1_Second
        MOVLW   OUTER_MAX_PLUS_1
        MOVWF   OUTER_DELAY
        MOVLW   REMAINDER
        MOVWF   REMAINDER_DELAY
Inner_Loop
        DECFSZ  INNER_DELAY, f
        GOTO    Inner_Loop
        MOVLW   INNER_MAX_PLUS_1
        MOVWF   INNER_DELAY
Middle_Loop
        DECFSZ  MIDDLE_DELAY, f
        GOTO    Inner_Loop
        MOVLW   MIDDLE_MAX_PLUS_1
        MOVWF   MIDDLE_DELAY
Outer_Loop
        DECFSZ  OUTER_DELAY, f
        GOTO    Inner_Loop
Remainder_Loop
        DECFSZ  REMAINDER_DELAY, f
        GOTO    Remainder_Loop
        RETURN

;------------Linear XOR Cascade Function-------------------------------;
Linear_Xor_Function     MACRO   LFS_Register, Reg_Size, Tap_Mask
        ;------ Pass the function arguments by Copy---------- ------;
        local i = 0
        while i < LFSR_SIZE
         MOVFW  (LFS_Register + i)      ; copy the LSFR to accumulator
         ANDLW  (Tap_Mask + i)          ; AND accum. copy with TAP_MASK
         MOVWF  (LOCAL_LFSR + i)        ; move from accum. to local
            function variable
         i += 1
        endw
        ;---------- Initialize Local Variables ---------------------;
        CLRF    PROP_SIGNAL             ; initialize XOR signal to 0
        MOVLW   (8 * Reg_Size)          ; start XOR adding at most
           significat bit
```

```
        BCF     STATUS, C               ; set initial XOR add input to
           0
Xor_Propagation_Loop
        ;---------- rotate 1 bit from local copy of LSFR ----------;
        RLF_Word        LOCAL_LFSR, LFSR_SIZE
        ;---------- Add Carry Bit to PROP_SIGNAL --------------------;
        MOVFW   STATUS          ; get status register for the carry bit
        ANDLW   1               ; keep only the carry bit
        ADDWF   PROP_SIGNAL, 1  ; add carry bit to prop signal
        DECFSZ  BIT_INDEX, 1    ; break loop after all bits propagated
        GOTO    Xor_Propagation_Loop
;-------Return (1s-Digit of PROP_SIGNAL) via Carry Flag ------------;
        RRF     PROP_SIGNAL, 1
        ENDM


;--------------------Initialize Data Memory-------------------------------;
Initialize
        ;---------- Initialize I/O
        BANKSEL TRISD           ; select Register Bank 1
        CLRF    TRISD           ; make Port D all output pins
        BANKSEL PORTD           ; back to Register Bank 0
        ;------------Initialize LSFRs-------------------------------------;
        MOVLW   INITIAL_VALUE
        MOVWF   LFSR1

;----------------Begin Main Program Loop-----------------------------------;
Main
        ;--------------Update LED Display-----------------------------------;
        MOVFW   LFSR1
        ANDLW   LED_MASK
        MOVWF   PORTD
        ;--------------------Cycle LSFR -------------------------------------;
        Linear_Xor_Function     LFSR1, LFSR_SIZE, LFSR1_TAP_MASK
                ; return value is in the Carry Flag
        RLF_Word        LFSR1, LFSR_SIZE        ; shift LSFR with
           feedback from STATUS, C
        ;----------------Delay 1 Second---------------------------------;
        CALL    Pause_1_Second
        GOTO    Main

        END
```

8.2 Alternating Step Generator

```
; asg.asm
; 3-LFSR Alternating Step Generator on PIC Development Board
; Ben Lorenzetti
; Embedded Systems Design, Fall 2015

#include <p16f887.inc>
        __CONFIG        _CONFIG1, _LVP_OFF & _FCMEN_OFF & _IESO_OFF &
           _BOR_OFF & _CPD_OFF & _CP_OFF & _MCLRE_OFF & _PWRTE_ON &
           _WDT_OFF & _INTRC_OSC_NOCLKOUT
        __CONFIG        _CONFIG2, _WRT_OFF & _BOR21V

;------------LFSR Bit/Byte Sizes, Tap Locations, and Initial Values
   -------------;
#define LFSR_SIZE 1             ; LFSR byte size
#define LED_MASK        0x0F    ; 0x07 for 3 bit LFSR, 0x0F for 4-bit
   LFSR, 0xFF for 8-bit LFSR
#define LFSR1_TAP_MASK  0x0C    ; 0x06 for 3-bit LFSR, 0x0C for 4-bit
   LFSR, 0xB8 for 8-bit LFSR
#define INITIAL_VALUE   1

;-------------------------Organize Program Memory-------------------------;
Reset_Vector
        ORG 0
        GOTO Initialize

Interupt_Vector
        ORG 4

;---------------------Allocate Static Variables-------------------------;
        cblock  0x20
        OUTER_DELAY
        MIDDLE_DELAY
        INNER_DELAY
        REMAINDER_DELAY
        PROP_SIGNAL
        BIT_INDEX
        LOCAL_LFSR: LFSR_SIZE
        LFSR1: LFSR_SIZE
        endc

;-------- void rotate_word_left (word*, word_size); ------------------;
RLF_Word        MACRO   Word_Addr, Word_Size
        local i = 0
        while i < Word_Size
          RLF   (Word_Addr + i), 1
          i += 1
```

```
        endw
        ENDM

#define OUTER_MAX_PLUS_1 60
#define MIDDLE_MAX_PLUS_1 40
#define INNER_MAX_PLUS_1 40
#define REMAINDER 1
;---------------------- void Pause_1_Second () -------------------------;
Pause_1_Second
        MOVLW   OUTER_MAX_PLUS_1
        MOVWF   OUTER_DELAY
        MOVLW   REMAINDER
        MOVWF   REMAINDER_DELAY
Inner_Loop
        DECFSZ  INNER_DELAY, f
        GOTO    Inner_Loop
        MOVLW   INNER_MAX_PLUS_1
        MOVWF   INNER_DELAY
Middle_Loop
        DECFSZ  MIDDLE_DELAY, f
        GOTO    Inner_Loop
        MOVLW   MIDDLE_MAX_PLUS_1
        MOVWF   MIDDLE_DELAY
Outer_Loop
        DECFSZ  OUTER_DELAY, f
        GOTO    Inner_Loop
Remainder_Loop
        DECFSZ  REMAINDER_DELAY, f
        GOTO    Remainder_Loop
        RETURN

;------------Linear XOR Cascade Function-------------------------------;
Linear_Xor_Function     MACRO   LFS_Register, Reg_Size, Tap_Mask
        ;------ Pass the function arguments by Copy---------- ------;
        local i = 0
        while i < LFSR_SIZE
         MOVFW  (LFS_Register + i)      ; copy the LSFR to accumulator
         ANDLW  (Tap_Mask + i)          ; AND accum. copy with TAP_MASK
         MOVWF  (LOCAL_LFSR + i)        ; move from accum. to local
            function variable
         i += 1
        endw
        ;---------- Initialize Local Variables ---------------------;
        CLRF    PROP_SIGNAL             ; initialize XOR signal to 0
        MOVLW   (8 * Reg_Size)          ; start XOR adding at most
           significat bit
        BCF     STATUS, C               ; set initial XOR add input to
           0
```

```
Xor_Propagation_Loop
        ;---------- rotate 1 bit from local copy of LSFR ----------;
        RLF_Word        LOCAL_LFSR, LFSR_SIZE
        ;---------- Add Carry Bit to PROP_SIGNAL --------------------;
        MOVFW   STATUS          ; get status register for the carry bit
        ANDLW   1               ; keep only the carry bit
        ADDWF   PROP_SIGNAL, 1  ; add carry bit to prop signal
        DECFSZ  BIT_INDEX, 1    ; break loop after all bits propagated
        GOTO    Xor_Propagation_Loop
;-------Return (1s-Digit of PROP_SIGNAL) via Carry Flag ------------;
        RRF     PROP_SIGNAL, 1
        ENDM


;--------------------Initialize Data Memory-------------------------------;
Initialize
        ;---------- Initialize I/O
        BANKSEL TRISD           ; select Register Bank 1
        CLRF    TRISD           ; make Port D all output pins
        BANKSEL PORTD           ; back to Register Bank 0
        ;------------Initialize LSFRs-------------------------------------;
        MOVLW   INITIAL_VALUE
        MOVWF   LFSR1

;----------------Begin Main Program Loop-------------------------------;
Main
        ;--------------Update LED Display-------------------------------;
        MOVFW   LFSR1
        ANDLW   LED_MASK
        MOVWF   PORTD
        ;--------------------Cycle LSFR ------------------------------;
        Linear_Xor_Function     LFSR1, LFSR_SIZE, LFSR1_TAP_MASK
                ; return value is in the Carry Flag
        RLF_Word        LFSR1, LFSR_SIZE        ; shift LSFR with
           feedback from STATUS, C
        ;----------------Delay 1 Second-----------------------------;
        CALL    Pause_1_Second
        GOTO    Main

        END
```