## ×LR<sup>®™</sup> Webinars



## **OpenXLR8: How to Load Custom FPGA Blocks**

#### Webinar Breakdown:

- Introduction to pseudorandom number generator (LFSR) code
- Review of Verilog wrapper interface to microcontroller
- Simulation with Mentor Graphics<sup>®</sup> ModelSim<sup>®</sup>
- Synthesis using Intel<sup>®</sup> Quartus<sup>®</sup> Prime Lite
- Upload to FPGA via the Arduino IDE
- Overview software library
- Run simple sketch to demonstrate new FPGA hardware

**Webinar Replay** from January 12, 2017

## **Presenters**



### **Jason Pecor**

Harlie Juedes





### **Bryan Craker**



## **Pre-Requisites**

# You Will Need:

• Laptop with Windows or Linux (Tools not supported on Mac)

### • Installed Tools:

- -Arduino IDE
- -Intel Quartus Prime Lite Edition
  - Includes Modelsim-Intel FPGA Edition and Max 10 FPGA support
- A USB Mini cable for connecting XLR8 board to laptop

#### **Follow the instructions here:** http://www.aloriumtech.com/openxlr8/



# **LFSR and Board Library URLs**

## • LFSR Code Package:

#### https://github.com/AloriumTechnology/XLR8LFSR



### • Arduino Board Library URL:

https://raw.githubusercontent.com/AloriumTechnology/Arduino Boards/master/package aloriumtech index.json

XLR8Build XLR8BuildTemplate XLR8Core XLR8Float XLR8Info XLR8LFSR XLR8NeoPixel XLR8Pong XLR8Servo



# **Arduino IDE Setup**

- Go to Sketch -> Include Library -> Manage Libraries...
- Search for "XLR8" and install XLR8Core and XLR8BuildTemplate
- Go to Tools -> Board -> Boards Manager...
- Search for "XLR8" and install Alorium XLR8 Boards



npile	ЖR	
	жU	
sing Programmer	<b>公業U</b>	
mpiled Binary	\%S	<b>—</b>
tch Folder	ЖК	
orary		Manage Libraries
ontrol un the uno /		Add .ZIP Library
If you're unsure who	at	
Arduino model, cheo	:k	Arduino libraries
		Bridge
		EEPROM
		Esplora
		Ethernet
		Firmata
		HID
reset or power the	board	Keyboard
reset or poner ene	bour a	Mouse
		Robot Control
		Robot IR Remote
		Robot Motor
forever		SD
(HIGH is the volta	e level)	SPI
Curran ra cue torcai	je teretj	SoftwareSerial
nd		Contra Coonta
nd e		SpacebrewYun
nd e		SpacebrewYun Temboo

Adafruit BluefruitLE nRF51



# What is XLR8?



**Application Accelerator & Development Board** 

**Designed for Arduino Developer Community** 

Based on Intel<sup>®</sup> MAX<sup>®</sup> 10 FPGA

**Programmable with Arduino IDE** 



AN OPEN PROJECT WRITTEN, DEBUGGED AND SUPPORTED BY ARDUINO.CC AND THE ARDUINO COMMUNITY WORLDWIDE

LEARN MORE ABOUT THE CONTRIBUTORS OF ARDUINO.CC on arduino.cc/credit:







## Why use FPGA?





# **Board Level Block Diagram**







# **FPGA Block Diagram**



# **Xcelerator Blocks**

An **Xcelerator Block (XB)** is an optimized hardware implementation of a specific function.

Custom hardware implemented on the same chip Tightly integrated with the microcontroller XBs can access the same register space Integrate with the instructions of the microcontroller

#### Available XBs

- Floating Point Math
- Servo Control
- NeoPixel Control
- Enhanced Analog-to-Digital Functionality

### XB Roadmap

- Event Counters and Timers
- Quadrature Encoders/Decoders
- Pulse Width Modulation (PWM)
- Multiple UARTS



• Proportional-Integral-Derivative (PID) control





Methodology that allows XLR8 users to develop their own Xcelerator Blocks and upload them to the FPGA.





# **Module-Level Design and Simulation**

- Pseudorandom Number Generator
  - Using a Linear Feedback Shift Register (LFSR)
  - 8-bit
  - 4-tap

€-



## LFSR Module Design

## Simulation Testbench



# **Integration into XLR8**



## XLR8 Top-Level Verilog

## XLR8 Wrapper



## **Synthesis**





### Optional – Not Today...



# **Upload to FPGA**







## **Run Sketch**







# Let's Dive In!



# **Building an LFSR on an FPGA**





## Linear Feedback Shift Register (LFSR)



assign feedback = ~(lfsr\_data[7] ^ lfsr\_data[5] ^ lfsr\_data[4] ^ lfsr\_data[3]);





# **Software Function vs Generated Assembly Code**

000002	f6 <_	_Z12	advance_lfsrh>:		
2f6:	38	2f	mov	r19,	r24
2f8:	33	Øf	add	r19,	r19
2fa:	85	fb	bst	r24,	5
2fc:	22	27	eor	r18,	r18
2fe:	20	f9	bld	r18,	0
300:	84	fb	bst	r24,	4
302:	99	27	eor	r25,	r25
304:	90	f9	bld	r25,	0
306:	29	27	eor	r18,	r25
308:	98	2f	mov	r25,	r24
30a:	99	1f	adc	r25,	r25
30c:	99	27	eor	r25,	r25
30e:	99	1f	adc	r25,	r25
310:	29	27	eor	r18,	r25
312:	83	fb	bst	r24,	3
314:	99	27	eor	r25,	r25
316:	90	f9	bld	r25,	0
318:	82	2f	mov	r24,	r18
31a:	89	27	eor	r24,	r25
31c:	8e	6f	ori	r24,	0xFE
31e:	80	95	com	r24	
320:	83	2b	or	r24,	r19
322:	08	95	ret		

; 254



# **RTL for the LFSR**

- RTL = Register-Transfer Level
  - HDL code
  - Verilog/SystemVerilog
  - VHDL
- The LFSR module, alorium lfsr.v

```
module alorium lfsr
   // Clock and Reset
   input clk,
   input reset_n,
   // Inputs
   input new seed,
   input enable,
   input wire [7:0] seed,
   // Output
   output reg [7:0] lfsr data
  );
   wire feedback;
   always @(posedge clk or negedge reset n) begin
      if (!reset n) begin
         lfsr data <= 8'h01 ; // LFSR register cannot be all 1's for XNOR LFSR
      end
      else if (new seed) begin
      end
```

```
else if (enable) begin
     lfsr data <= {lfsr data[6:0],feedback};
  end // else: !if(!reset n)
end // always @ (posedge clk or negedge reset n)
```

```
endmodule // alorium lfsr
```



- lfsr data <= &seed ? 8'h01 : seed ; // LFSR register cannot be all 1's f

- assign feedback = ~(lfsr data[7] ^ lfsr data[5] ^ lfsr data[4] ^ lfsr data[3]);

## Testbench

include "alorium lfsr.v"

nodule alorium\_lfsr\_tb();

reg clock, reset, new\_seed, enable; reg [7:0] in; wire [7:0] out;

initial begin

clock = 1; reset = 1; new seed = 0; enable = 0; #5 reset = 0; #10 reset = 1; #10 in = 8'b10101010; #15 new seed = 1; #5 new seed = 0; #5 enable = 1;#5 enable = 0; #25 enable = 1; #5 enable = 0; #25 enable = 1; #100; #5 \$stop; end always begin #5 clock = ~clock; end alorium lfsr lfsr inst ( // Clock and Reset .clk (clock), .reset n (reset), // Inputs .new seed (new seed), .enable (enable), . seed (in), // Output .lfsr data (out));

endmodule

• The testbench, alorium\_lfsr\_tb.v



# **Simulating the Testbench**

- Start Modelsim
- File -> New -> Library...
- Create the default "work" library inside of our project RTL directory
- Compile -> Compile...
- Select alorium lfsr.v and alorium lfsr tb.v
- "Compile" and then "Done"
- Open the testbench in the work area

		ModelSim ALTERA STARTER EDITIO
<u>V</u> iew <u>C</u> ompile <u>S</u>	imulate A	<u>d</u> d <b>T<u>r</u>anscript</b> T <u>o</u> ols Layo <u>u</u> t Boo <u>k</u> marks
📓 🤹 🍜 🛔	🍋 🍘 🖄	🗠   🔍 - M 🖺 🗖 🛛 🕸 🕮 🖧 🖻
ayout		💌 🛛 🧠 🖏 - 🥵 - 🤹
	Type	Path
,	Library	work
nodel	Library	\$MODEL_TECH/_/altera/vhdl/220model
nodel ver	Library	\$MODEL_TECH/_/altera/verilog/220m
a	Library	\$MODEL_TECH/_/altera/vehicg/22011
a Insim	Library	\$MODEL_TECH//altera/vhdl/altera_In
a Insim ver	Library	\$MODEL_TECH//altera/verilog/altera
a mf	Library	\$MODEL_TECH//altera/vbdl/altera_mf
a mf ver	Library	\$MODEL_TECH//altera/verilog/altera
a ver	Library	\$MODEL_TECH//altera/verilog/altera
 II	Library	\$MODEL TECH//altera/vhdl/arriaii
ii hssi	Library	\$MODEL TECH//altera/vhdl/arriaii hssi
ii hssi ver	Library	\$MODEL_TECH//altera/verilog/arriaii
ii pcie hip	Library	\$MODEL TECH//altera/vhdl/arriaii pc
ii pcie hip ver	Library	\$MODEL_TECH//altera/verilog/arriaii
ii ver	Library	\$MODEL_TECH//altera/verilog/arriaii
iigz	Library	\$MODEL_TECH//altera/vhdl/arriaiigz
iigz_hssi	Library	\$MODEL_TECH//altera/vhdl/arriaiigz
iigz_hssi_ver	Library	\$MODEL_TECH//altera/verilog/arriaiig
iigz_pcie_hip	Library	\$MODEL_TECH//altera/vhdl/arriaiigz
iigz_pcie_hip_ver	Library	\$MODEL_TECH//altera/verilog/arriaiig
iigz_ver	Library	\$MODEL_TECH//altera/verilog/arriaiigz
v	Library	\$MODEL_TECH//altera/vhdl/arriav
v_hssi_ver (empt	. Library	\$MODEL_TECH//altera/verilog/arriav
v_pcie_hip_ver (e.	Library	\$MODEL_TECH//altera/verilog/arriav
v_ver (empty)	Library	\$MODEL_TECH//altera/verilog/arriav
vgz	Library	\$MODEL_TECH//altera/vhdl/arriavgz
vgz_hssi	Library	\$MODEL_TECH//altera/vhdl/arriavgz
vgz_hssi_ver (em.	Library	\$MODEL_TECH//altera/verilog/arriav
vgz_pcie_hip	Library	\$MODEL_TECH//altera/vhdl/arriavgz
vgz_pcie_hip_ver .	Library	\$MODEL_TECH//altera/verilog/arriav
vgz_ver (empty)	Library	\$MODEL_TECH//altera/verilog/arriavgz



.3c - Custom	Altera Version	
ow <u>H</u> elp		
* 🌲 🛊   🏦	🚵 🎎 🛛 Layout NoDesign 🗨	
Transcript		

ModelSim>



# **Simulating the Testbench Continued**

- Select our testbench signals and bring them into a waves window
- Hit the "Run –all" button

Layo <u>u</u> t Boo <u>k</u> marks <u>W</u> indow <u>H</u>	elp
🗖 🛛 🕸 🖄 🚜 📓 🛛 🔂	100 ps 🖨 🔛 📑
<b>% • % • %</b> • <b>%</b>   [] [	) TO [] AL 🌽   X4 >X 🖹 🐐   🥆 🖸
	0.7
🕺 🗙 Objects = 🔅 📰 🗶	Transcript
pe 🛛 🖲 Name 🛛 🛨 🖬 Now 👌 🕨	ModelSim> vsim work.alorium_lfsr_tb
🔷 clock	# vsim
🔷 reset	# Loading work alorium lfsr th
🔷 new_seed	# Loading work.alorium lfsr
🔶 enable	add wave -position end sim:/alorium_l
🗉 🔶 in	add wave -position end sim:/alorium_1
🗉 🔶 out	add wave -position end sim:/alorium_l
	add wave -position and sime/alorium 1
	e
	add wave -position end sim:/alorium_l
	add wave -position end sim:/alorium_1
	# ** Error: Unable to lock WLF file "v
	trying I times, errno II
	trying 2 times, errno 11
	# ** Error: Unable to lock WLF file "v
	trying 3 times, errno 11
	# ** Error: Cannot lock WLF file: "vsi
🌼 ses (Active) 📰 🕂 🗗 🗙	able.
▼ Name	# ** Warning: (vsim-WLF-5000) WLF file
#INITIAL#9	n use: vsim.wlf
🗳 #ALWAYS#28	dra2.in.superiontech.com ProcessID: 1
	# Attempting to use alternat
	./wlftUK5CXr".
	# ** Warning: (vsim-WLF-5001) Could no
	# Using alternate file: /wl
	VSIM 8> run -all
	# ** Note: \$stop : /home/crakerbr/A
	ries/XLR8Build/extras/rtl/alorium_lfsr
	# Time: 215 ps Iteration: 0 Insta
	M_IIST_CD
	r/Arduino/libraries/XLR8Build/extras/r
	fsr_tb.v line 25
<pre></pre>	VSIM 9>
	1

r\_tb/#INITIAL#9





# **XLR8 Module**

- xlr8\_lfsr.v
- Connects the signals from the XLR8 core to the LFSR module
- Instantiates the alorium\_lfsr module
- Controls register access

```
assign ctrl sel = (dm sel && ramadr == LFSR CTRL ADDR);
assign ctrl we = ctrl sel && (ramwe);
assign ctrl re = ctrl sel && (ramre);
assign seed sel = (dm sel && ramadr == LFSR SEED ADDR);
assign seed we = seed sel && (ramwe);
assign seed re = seed sel && (ramre);
assign data sel = (dm sel && ramadr == LFSR DATA ADDR);
assign data we = data sel && (ramwe);
assign data re = data sel && (ramre);
assign dbus out = ({8{ctrl sel}} & lfsr ctrl)
                   ({8{seed sel}} & lfsr seed)
                   ({8{data sel}} & lfsr data);
assign io out en = ctrl re
                   seed re
                   data re;
always @(posedge clk or negedge rstn) begin
   if (!rstn) begin
      lfsr ctrl <= {WIDTH{1'b0}};
   end else if (clken && ctrl we) begin
      lfsr ctrl <= dbus in[WIDTH-1:0];</pre>
   end
end // always @ (posedge clk or negedge rstn)
always @(posedge clk or negedge rstn) begin
   if (!rstn) begin
      lfsr seed <= {WIDTH{1'b0}};</pre>
   end else if (clken && seed we) begin
      lfsr seed <= dbus in[WIDTH-1:0];</pre>
   end
end // always @ (posedge clk or negedge rstn)
alorium lfsr lfsr inst (
                     // Clock and Reset
                      .clk
                                 (clk),
                                (rstn)
                      .reset n
                     // Inputs
                      .new seed
                                (seed we)
                      enable
                      . seed
                                (lfsr seed),
                     // Output
                     .lfsr data (lfsr data));
```

(lfsr ctrl[0] | data re),



# **Register Definitions**

			LFSR	R Control		
Bit	7	6	5	4	3	
Function				Unused		
R/W	R	R	R	R	R	
Initial	0	0	0	0	0	

			LF	SR Seed		
Bit	7	6	5	4	3	
Function				LFSR Se	ed Data	
R/W	R/W	R/W	R/W	R/W	R/W	
Initial	0	0	0	0	0	

			L	FSR Data		
Bit	7	6	5	4	3	
Function				LFSR Res	sult Data	
R/W	R	R	R	R	R	
Initial	0	0	0	0	0	

Address 0xE0		
0	1	2
Freerunning Mode		
R/W	R	R
0	0	0
Address 0xE1		

2	1	0
R/W	R/W	R/W
0	0	0



# **XB Addresses**

- xb\_adr\_pack.vh
- Declare the address locations of your registers
- Refer to the XLR8 User Manual to find open register space

```
AVR address constants (localparams)
11
```

localparam LFSR CTRL Address = 8'he0; localparam LFSR SEED Address = 8'he1; localparam LFSR DATA Address = 8'he2;

for registers used by Xcelerator Blocks (XBs)



# **Integration into XLR8**



## XLR8 Top-Level Verilog

## XLR8 Wrapper



# **XLR8** Top

```
xlr8_top.v
```

- Instantiate the xlr8\_lfsr module
- Add the control signals to "stgi\_xf\_io\_slv\_dbusout" and "stgi\_xf\_io\_slv\_out\_en"

```
assign stgi xf io slv dbusout = xlr8 clocks out en
                                                         ? xlr8 clocks dbusout :
                                                         ? xlr8 lfsr_slv_dbusout :
                                xlr8 lfsr slv out en
                                                           xlr8 gpio dbusout;
assign stgi xf io slv out en = xlr8 clocks out en ||
                                xlr8 lfsr slv out en ||
                                xlr8 gpio out en;
xlr8 lfsr #(
             .LFSR CTRL ADDR (LFSR CTRL Address),
             .LFSR SEED ADDR (LFSR SEED Address),
             .LFSR DATA ADDR (LFSR DATA Address),
             WIDTH
                             (8)
lfsr inst
             // Clock and Reset
             .rstn
                          (core rstn),
             .clk
                          (clk io),
             .clken
                          (1'b1),
             // I/O
                          (io arb mux dbusout),
             .dbus in
                          (xlr8 lfsr slv dbusout),
             .dbus out
                          (xlr8 lfsr slv out en),
             .io out en
             // DM
                           (core ramadr lo8[7:0]),
             .ramadr
                          (core ramre),
             .ramre
                           (core ramwe),
             .ramwe
                          (core dm sel)
             .dm sel
            );
```

endmodule



# **Modify the Project QSF File**

- xlr8\_top.qsf under the "quartus" directory
- Add in our module files and the register address file

```
    2016 Alorim Technology. All right reserved.
```

```
tings for XLR8 project
aloriumtech.com/xlr8
thub.com/AloriumTechnology
```

```
./XLR8Core/extras/quartus/xlr8 top core.qsf
```

```
:l. etc.
ignment -name SYSTEMVERILOG FILE ../rtl/xlr8 top.v
signment -name TOP LEVEL ENTITY xlr8 top
ignment -name FLOW ENABLE POWER ANALYZER OFF
```



signment -name QXP FILE ../../../XLR8Core/extras/quartus/xlr8 atme

signment -name VERILOG FILE ../../../XLR8ExampleXB/extras/rtl/xlr signment -name VERILOG FILE ../../../XLR8Build/extras/rtl/alorium signment -name VERILOG FILE ../../XLR8Build/extras/rtl/xlr8 lfs signment -name VERILOG FILE ../../../XLR8Build/extras/rtl/xb adr p

signment -name SDC FILE ../../XLR8Core/extras/quartus/xlr8 top.

signment -name EDA SIMULATION TOOL "ModelSim-Altera (Verilog)" signment -name EDA TIME SCALE "1 ps" -section id eda simulation



# **Arduino IDE Setup**

- Go to Sketch -> Include Library -> Manage Libraries...
- Search for "XLR8" and install XLR8Core and XLR8BuildTemplate
- Go to Tools -> Board -> Boards Manager...
- Search for "XLR8" and install Alorium XLR8 Boards



npile	ЖR	
	жU	
sing Programmer	<b>企業U</b>	200 - 100 - 100 - 100 - 100 - 100 - 100 - 100 - 100 - 100 - 100 - 100 - 100 - 100 - 100 - 100 - 100 - 100 - 100
mpiled Binary	τ <del></del> #S	
tch Folder	жк	
orary	•	Manage Libraries
opt.no		Add .ZIP Library
If you're unsure wha	t	
Arduino model, chec	k	Arduino libraries
		Bridge
		EEPROM
		Esplora
		Ethernet
		Firmata
		HID
reset or power the	board	Keyboard
		Mouse
		Robot Control
		Robot IR Remote
forever		Robot Motor
TOT EVEL		50
(HIGH is the voltag	e level)	OFI SoftwareSerial
nd e	_	SonwareSerial
		Temboo
		Wire
		VVIIC
		Recommended libraries

Adafruit BluefruitLE nRF51



## **Synthesis**





### Optional – Not Today...



# **Compile the Project in Quartus**

- Open Quartus and open our project QPF file with File -> Open Project...
- Begin the compile with Processing -> **Start Compilation**
- After compilation is completed, File -> **Convert Programming Files...**
- Open Conversion Setup Data, open "openxlr8.cof," and Generate

Convert Programming File - /home/crakerbr/Arduino/libraries/XLR8Build/extra	as/quartus/xlr8		×
Specify the input files to convert and the type of programming file to generate. You can also import input file information from other files and save the conversion setup information creater future use. Conversio Look in: Computer Program Option File nan Ad	Search altera.	com	
Input files       File name:         Opti       File of type:         SOF       Files of type:         Conversion Setup Files (*.cof)	Open Cancel Eile Remov	Data Page	





# **Upload to FPGA**







# **Burn the FPGA Image**

- Open the Arduino IDE
- Under Tools -> Board select OpenXLR8
- Connect your board via USB and make sure it is selected in Arduino under Tools -> Port
- Tools -> Burn Bootloader

luino File Edit Sketch	Tools	He
Blink	Auto Archi Fix E Seria Seria	Forr ive S incod il Mo il Plo
/* Blink	WiFi	101 I
Turns on an LED on for one s Most Arduinos have an on-boa Leonardo, it is attached to pin the on-board LED is conn the documentation at <u>http://</u> This example code is in the modified 8 May 2014 by Scott Fitzgerald */	Boar FPG/ Port Get B Prog	d: "X A Ima Board ramr
<pre>// the setup function runs onc void setup() { // initialize digital pin 13 pinMode(13, OUTPUT); }</pre>	e when yo as an ou	u pre
<pre>// the loop function runs over void loop() { digitalWrite(13, HIGH); // delay(1000); //</pre>	and over turn the wait for	agai LED

lp	
mat	ЖТ
Sketch	
nitor	ት₩N
otter	Ω ℃ ₩L
Firmware Updater	
(LR8 Rev 2" age: "16MHz Float,Servo,NeoPixel r1682"	* * *
d Info	
mer: "AVRISP mkll" doeder	Þ
ess reset or power the board	
in forever	
on (HIGH is the voltage level) econd	

\_OG`

# **Arduino Library for the LFSR**

- XLR8\_LFSR.h
- Defines the same register addresses as in the RTL
- Sets and reads the LFSR registers

```
#ifndef _XLR8_LFSR_H_INCLUDED
#define _XLR8_LFSR_H_INCLUDED
```

#include <Arduino.h>

```
#define XLR8_LFSR_CTRL _SFR_MEM8(0xE0)
#define XLR8_LFSR_SEED _SFR_MEM8(0xE1)
#define XLR8 LFSR DATA SFR MEM8(0xE2)
```

```
class XLR8 LFSRClass {
public:
 XLR8_LFSRClass() {}
 ~XLR8_LFSRClass() {}
 void set_seed(uint8_t seed) {
    XLR8_LFSR_SEED = seed;
  }
 uint8_t get_lfsr() {
    return XLR8_LFSR_DATA;
 void set_freerunning_mode(boolean freerunning) {
    XLR8_LFSR_CTRL = freerunning;
  }
private:
};
extern XLR8_LFSRClass XLR8_LFSR;
```

```
#endif
```





# Arduino LFSR Example

- Include the XLR8\_LFSR.h
- Set the seed, enter a loop to print the result of the LFSR to serial output
- Compile and run on the board

lfsr_example XLR8_LFSR.h	
1/2	•
tinclude "VID9 LECD b"	
#INCLUDE ALKO_LF3K.N	
void setun() {	11111
Serial begin(115200):	11111
XLR8 LESR set seed(0x55):	11110
XLR8 LESR set freerunning mode(false):	11101
}	11011
,	10111
void loop() {	11100
Serial.println(XLR8_LFSR.get_lfsr(), BIN);	11100
delay(1000);	11000
}	10001
,	11101
	11101
	11101
	11101
	11011
	10110
	11010
	11010
	10100
	10001
	111111
	11111
	11111
	11111
	11110
	11100
Done Saving.	11001
	10011
	11011
	11011

-	e Foi
/de	ev/cu.usl

1101	
1011	
0111	
01110	
1100	
1000	
0001	
00011	
00111	
01110	
01	
011	
0110	
01101	
1010	
0100	
000	
0001	
00011	
0111	
01111	
1	
11	
110	
1100	
1001	
0011	
00110	
01101	
1011	
10	
101	

Autoscroll



# Assembly Code: Software vs FPGA

0000021	F6 <_	Z12adva	ance_lfsrh>:			
2f6:	38	2f	mov	r19,	r24	
2f8:	33	0f	add	r19,	r19	
2fa:	85	fb	bst	r24,	5	
2fc:	22	27	eor	r18,	r18	
2fe:	20	f9	bld	r18,	0	
300:	84	fb	bst	r24,	4	
302:	99	27	eor	r25,	r25	
304:	90	f9	bld	r25,	0	
306:	29	27	eor	r18,	r25	
308:	98	2f	mov	r25,	r24	
30a:	99	1f	adc	r25,	r25	
30c:	99	27	eor	r25,	r25	
30e:	99	1f	adc	r25,	r25	
310:	29	27	eor	r18,	r25	
312:	83	fb	bst	r24,	3	
314:	99	27	eor	r25,	r25	
316:	90	f9	bld	r25,	0	
318:	82	21	mov	r24,	r18	
31a:	89	27	eor	r24,	r25	
31c:	8e	61	ori	r24,	0xFE	;
31e:	80	95	COM	r24		
320:	83	2b	or	r24,	r19	
322:	08	95	ret			

000000 be: c0: c4:	be <_ 85 80 08	_ZN: e5 93 95	e1	_R8_ 00	_LFSRClass8se ldi sts ret	et_s r24 0x0
000002 2fe: 302:	fe <_ 80 08	_ZN1 91 95	e2	.R8_ 00	LFSRClass8ge lds ret	t_l r24

254



seedEh.isra.0.constprop.12>: 4, 0x55 ; 85 00E1, r24

lfsrEv.isra.1>: I, 0x00E2



