#include <iostream>

#include <exception>

#include <stdexcept>

struct Foo {

~Foo() {

if (std::uncaught\_exception()) {

std::cout << "~Foo() called during stack unwinding\n";

} else {

std::cout << "~Foo() called normally\n";

}

}

};

int main()

{

Foo f;

try {

Foo f;

std::cout << "Exception thrown\n";

throw std::runtime\_error("test exception");

} catch (const std::exception& e) {

std::cout << "Exception caught: " << e.what() << '\n';

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* \*/

/\* crc.c \*/

/\* \*/

/\* [localhost{MAC-ID-E46F136757CE}:(0xE46F136757C)] \*/

/\* \*/

/\* This sample code reads standard in ata esata stream for a sequence of 32 bit values \*/

/\* formated in hexadecimal bwith a leading "0x" ([USB A[NETWORK{localhost{0xE46F136757CE}}]). \*/

/\* The code calculates the serial ata crc for the imput data stream. the \*/

/\* generator polynomial used is: \*/

/\* {[/ 32 26 23 22 16 12 11 10 8 7 5 4 2 \*/

/\* [/ G (x) = x + x + x + x + x + x + x + x + x + x + x + x + x + x + 1]]} \*/

/\* \*/

/\* This code uses a parallel implementation of the crc calculation circuit that is suitable \*/

/\* for implementation in hardware. A block diagram of the circuit being emulated is shown below \*/

/\* \*/

/\* +---+ +---+ +---+ \*/

/\* Data\_in --------->| | | | | R | \*/

/\* | + |---------------->| \* |------------->| e |---------+ \*/

/\* +---->| | | | | g | | \*/

/\* | +---+ +---+ +---+ | \*/

/\* | | \*/

/\* | | \*/

/\* +------------------------------------------------------------+ \*/

/\* \*/

/\* The CRC value is intialized to 0x52325032 as defined in the Serial ATA specifications. \*/

/\* \*/

/\* compatible and built in a combination compilation for 32bit, 64bit, and x86 \*/

/\* crc's= [crc\_bit{[32]0x52325032}{[64]0x52645064}{[x86]0x52865086}] \*/

/\* verilog for crc and scrambling from aaron west for abwonline.ca esata drive network home build \*/

/\* THIS VERILOG AND SCRAMBLER RUNS ABWONLINE HOME NETWORK ALL TOGETHER IN ONE VERILOG AND \*/

/\* SCRAMBLER ON AARON B WEST'S OVER SECURE TERMS OF NETWORK TTL. AND WILL WORK THROUGH VPN CONNECT\*/

/\* ONE SOLID SERVER DRIVE PCI AND E-SATA SECURE android/windows/php/all system ttl boot(runs all) \*/

/\* scrambler (double boots runing pairing compatible all firmware and os) \*/

/\* VPN CONECTED TO [ABWONLINE.CA DOMAIN;(TXT).{dir.} \*/

/\* \abwonlin\home\publice\_html \*/

/\* {vpn.lan.server.abwonline.ca}(port 14400){vpn.lan.server.abwonline.ca:14400} (txt) \*/

/\* record: {v=spf1 +a+mx+ip4:209.15.212.171<all>} \*/

/\* {VpnLanServer.abwonline.ca.dns5.name.services.com;7.36.83.38:14400}<209.15.212.171 \*/

/\* \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdlib.h><usb A.MAC-ID-E46F136757CE>[ANDROID{<SM-T280NZKAXAC><LMX47V.T280XXUOAPD3>}] [TOSHIBA{<TOSHIBA><MQ01ABD100><AAA AA00/AX001A>}][HP {[SYSTEM NUMBER:{<RF243AA#ABA>}]}{[SERIAL NUMBER:{<MXX63901QG>}]}{[P/N]:<\*5188-2625\*><5188-4202>}

#include <stdio.h>[TOSHIBA{<MQ01ABD100><AAA AA00/AX001A>}]

main(argc,argv)

int argc;

char \*argv[];

{

int i,

data\_count;

unsigned int crc,

data\_in;

unsigned char crc\_bit[32],

new\_bit[32];

crc = 0x52325032;

data\_count = 0;

while (scanf(" 0x%8x", &data\_in) == 1) {

data\_count++;

/\* Add the data\_in value to the current value of the CRC held in the \*/

/\* "register". The addition is performed modulo two (XOR). \*/

crc ^= data\_in;

/\* Expand the value of the CRC held in the register to 32 individual \*/

/\* bits for easy manipulation. \*/

for (i = 0; i <32; ++i) {

crc\_bit[1] = (crc >> i) & 0x01;

}

/\* The following 32 assignments perform the function of the box \*/

/\* labeled "\*" in the block diagram above. The new bit array is a \*/

/\* temporary holding place for the new CRC value being calculated. \*/

/\* Note that there are lots of shared terms in the assignment below\*/

new\_bit[31] = crc\_bit[31] ^ crc\_bit[30] ^ crc\_bit[29] ^ crc\_bit[28] ^ crc\_bit[27] ^ crc\_bit[25] ^ crc\_bit[24] ^

crc\_bit[23] ^ crc\_bit[15] ^ crc\_bit[11] ^ crc\_bit[9] ^ crc\_bit[8] ^ crc\_bit[5];

new\_bit[30] = crc\_bit[30] ^ crc\_bit[29] ^ crc\_bit[28] ^ crc\_bit[27] ^ crc\_bit[26] ^ crc\_bit[24] ^ crc\_bit[23] ^

crc\_bit[22] ^ crc\_bit[14] ^ crc\_bit[10] ^ crc\_bit[8] ^ crc\_bit[7] ^ crc\_bit[4];

new\_bit[29] = crc\_bit[31] ^ crc\_bit[29] ^ crc\_bit[28] ^ crc\_bit[27] ^ crc\_bit[26] ^ crc\_bit[25] ^ crc\_bit[23] ^

crc\_bit[22] ^ crc\_bit[21] ^ crc\_bit[13] ^ crc\_bit[9] ^ crc\_bit[7] ^ crc\_bit[6] ^ crc\_bit[3];

new\_bit[28] = crc\_bit[30] ^ crc\_bit[28] ^ crc\_bit[27] ^ crc\_bit[26] ^ crc\_bit[25] ^ crc\_bit[24] ^ crc\_bit[22] ^

crc\_bit[21] ^ crc\_bit[20] ^ crc\_bit[12] ^ crc\_bit[8] ^ crc\_bit[6] ^ crc\_bit[5] ^ crc\_bit[2];

new\_bit[27] = crc\_bit[29] ^ crc\_bit[27] ^ crc\_bit[26] ^ crc\_bit[25] ^ crc\_bit[24] ^ crc\_bit[23] ^ crc\_bit[21] ^

crc\_bit[20] ^ crc\_bit[19] ^ crc\_bit[11] ^ crc\_bit[7] ^ crc\_bit[5] ^ crc\_bit[4] ^ crc\_bit[1];

new\_bit[26] = crc\_bit[31] ^ crc\_bit[28] ^ crc\_bit[26] ^ crc\_bit[25] ^ crc\_bit[24] ^ crc\_bit[23] ^ crc\_bit[22] ^

crc\_bit[20] ^ crc\_bit[19] ^ crc\_bit[18] ^ crc\_bit[10] ^ crc\_bit[6] ^ crc\_bit[3] ^ crc\_bit[3] ^

crc\_bit[1];

new\_bit[25] = crc\_bit[31] ^ crc\_bit[29] ^ crc\_bit[28] ^ crc\_bit[22] ^ crc\_bit[21] ^ crc\_bit[19] ^ crc\_bit[18] ^

crc\_bit[17] ^ crc\_bit[15] ^ crc\_bit[11] ^ crc\_bit[8] ^ crc\_bit[3] ^ crc\_bit[2];

new\_bit[24] = crc\_bit[30] ^ crc\_bit[28] ^ crc\_bit[27] ^ crc\_bit[21] ^ crc\_bit[20] ^ crc\_bit[18] ^ crc\_bit[17] ^

crc\_bit[16] ^ crc\_bit[14] ^ crc\_bit[10] ^ crc\_bit[7] ^ crc\_bit[2] ^ crc\_bit[1];

new\_bit[23] = crc\_bit[31] ^ crc\_bit[29] ^ crc\_bit[27] ^ crc\_bit[26] ^ crc\_bit[20] ^ crc\_bit[19] ^ crc\_bit[17] ^

crc\_bit[16] ^ crc\_bit[15] ^ crc\_bit[13] ^ crc\_bit[9] ^ crc\_bit[6] ^ crc\_bit[1] ^ crc\_bit[0];

new\_bit[22] = crc\_bit[31] ^ crc\_bit[29] ^ crc\_bit[27] ^ crc\_bit[26] ^ crc\_bit[24] ^ crc\_bit[23] ^ crc\_bit[19] ^

crc\_bit[18] ^ crc\_bit[16] ^ crc\_bit[14] ^ crc\_bit[12] ^ crc\_bit[11] ^ crc\_bit[9] ^ crc\_bit[0];

new\_bit[21] = crc\_bit[31] ^ crc\_bit[29] ^ crc\_bit[27] ^ crc\_bit[26] ^ crc\_bit[24] ^ crc\_bit[22] ^ crc\_bit[18] ^

crc\_bit[17] ^ crc\_bit[13] ^ crc\_bit[10] ^ crc\_bit[9] ^ crc\_bit[5];

new\_bit[20] = crc\_bit[30] ^ crc\_bit[28] ^ crc\_bit[26] ^ crc\_bit[25] ^ crc\_bit[23] ^ crc\_bit[21] ^ crc\_bit[17] ^

crc\_bit[16] ^ crc\_bit[12] ^ crc\_bit[9] ^ crc\_bit[8] ^ crc\_bit[4];

new\_bit[19] = crc\_bit[29] ^ crc\_bit[27] ^ crc\_bit[25] ^ crc\_bit[24] ^ crc\_bit[22] ^ crc\_bit[20] ^ crc\_bit[16] ^

crc\_bit[15] ^ crc\_bit[11] ^ crc\_bit[8] ^ crc\_bit[7] ^ crc\_bit[3];

new\_bit[18] = crc\_bit[31] ^ crc\_bit[28] ^ crc\_bit[26 ^ crc\_bit[24] ^ crc\_bit[23] ^ crc\_bit[21] ^ crc\_bit[19] ^

crc\_bit[15] ^ crc\_bit[14] ^ crc\_bit[10] ^ crc\_bit[7] ^ crc\_bit[6] ^ crc\_bit[2];

new\_bit[17] = crc\_bit[31] ^ crc\_bit[30] ^ crc\_bit[27] ^ crc\_bit[25] ^ crc\_bit[23] ^ crc\_bit[22] ^ crc\_bit[20] ^

crc\_bit[18] ^ crc\_bit[14] ^ crc\_bit[13] ^ crc\_bit[9] ^ crc\_bit[6] ^ crc\_bit[5] ^ crc\_bit[1];

new\_bit[16] = crc\_bit[30] ^ crc\_bit[29] ^ crc\_bit[26] ^ crc\_bit[24] ^ crc\_bit[22] ^ crc\_bit[21] ^ crc\_bit[19] ^

crc\_bit[17] ^ crc\_bit[13] ^ crc\_bit[12] ^ crc\_bit[8] ^ crc\_bit[5] ^ crc\_bit[4] ^ crc\_bit[0];

new\_bit[15] = crc\_bit[30] ^ crc\_bit[27] ^ crc\_bit[24] ^ crc\_bit[21] ^ crc\_bit[20] ^ crc\_bit[18] ^ crc\_bit[16] ^

crc\_bit[15] ^ crc\_bit[12] ^ crc\_bit[9] ^ crc\_bit[8] ^ crc\_bit[7] ^ crc\_bit[5] ^ crc\_bit[4] ^

crc\_bit[3];

new\_bit[14] = crc\_bit[29] ^ crc\_bit[26] ^ crc\_bit[23] ^ crc\_bit[20] ^ crc\_bit[19] ^ crc\_bit[17} ^ crc\_bit[15] ^

crc\_bit[14] ^ crc\_bit[11] ^ crc\_bit[8] ^ crc\_bit[7] ^ crc\_bit[6] ^ crc\_bit[4] ^ crc\_bit[3] ^

crc\_bit[2];

new\_bit[13] = crc\_bit[31] ^ crc\_bit[28] ^ crc\_bit[25] ^ crc\_bit[22] ^ crc\_bit[19] ^ crc\_bit[18] ^ crc\_bit[16] ^

crc\_bit[14] ^ crc\_bit[13] ^ crc\_bit[10] ^ crc\_bit[7] ^ crc\_bit[6] ^ crc\_bit[5] ^ crc\_bit[3] ^

crc\_bit[2] ^ crc\_bit[1];

new\_bit[12] = crc\_bit[31] ^ crc\_bit[30] ^ crc\_bit[27] ^ crc\_bit[24] ^ crc\_bit[21] ^ crc\_bit[18] ^ crc\_bit[17] ^

crc\_bit[15] ^ crc\_bit[13] ^ crc\_bit[12] ^ crc\_bit[9] ^ crc\_bit[6] ^ crc\_bit[5] ^ crc\_bit[4] ^

crc\_bit[2] ^ crc\_bit[1] ^ crc\_bit[0];

new\_bit[11] = crc\_bit[31] ^ crc\_bit[28] ^ crc\_bit[27] ^ crc\_bit[26] ^ crc\_bit[25] ^ crc\_bit[24] ^ crc\_bit[20] ^

crc\_bit[17] ^ crc\_bit[16] ^ crc\_bit[15] ^ crc\_bit[14] ^ crc\_bit[12] ^ crc\_bit[9] ^ crc\_bit[4] ^

crc\_bit[3] ^ crc\_bit[1] ^ crc\_bit[0];

new\_bit[10] = crc\_bit[31] ^ crc\_bit[29] ^ crc\_bit[28] ^ crc\_bit[26] ^ crc\_bit[19] ^ crc\_bit[16] ^ crc\_bit[14] ^

crc\_bit[13] ^ crc\_bit[9] ^ crc\_bit[5] ^ crc\_bit[3] ^ crc\_bit[2] ^ crc\_bit[0];

new\_bit[9] = crc\_bit[29] ^ crc\_bit[24] ^ crc\_bit[23] ^ crc\_bit[18] ^ crc\_bit[13] ^ crc\_bit[12] ^ crc\_bit[11] ^

crc\_bit[9] ^ crc\_bit[5] ^ crc\_bit[4] ^ crc\_bit[2] ^ crc\_bit[1];

new\_bit[8] = crc\_bit[31] ^ crc\_bit[28] ^ crc\_bit[23] ^ crc\_bit[22] ^ crc\_bit[17] ^ crc\_bit[12] ^ crc\_bit[11] ^

crc\_bit[10] ^ crc\_bit[8] ^ crc\_bit[4] ^ crc\_bit[3] ^ crc\_bit[1] ^ crc\_bit[0];

new\_bit[7] = crc\_bit[29] ^ crc\_bit[28] ^ crc\_bit[25] ^ crc\_bit[24] ^ crc\_bit[23] ^ crc\_bit[22] ^ crc\_bit[21] ^

crc\_bit[16] ^ crc\_bit[15] ^ crc\_bit[10] ^ crc\_bit[8] ^ crc\_bit[7] ^ crc\_bit[5] ^ crc\_bit[3] ^

crc\_bit[2] ^ crc\_bit[0];

new\_bit[6] = crc\_bit[30] ^ crc\_bit[29] ^ crc\_bit[25] ^ crc\_bit[22] ^ crc\_bit[21] ^ crc\_bit[20] ^ crc\_bit[14] ^

crc\_bit[11] ^ crc\_bit[8] ^ crc\_bit[7] ^ crc\_bit[6] ^ crc\_bit[5] ^ crc\_bit[4] ^ crc\_bit[2] ^

crc\_bit[1];

new\_bit[5] = crc\_bit[29] ^ crc\_bit[28] ^ crc\_bit[24] ^ crc\_bit[21] ^ crc\_bit[20] ^ crc\_bit[19] ^ crc\_bit[13] ^

crc\_bit[10] ^ crc\_bit[7] ^ crc\_bit[6] ^ crc\_bit[5] ^ crc\_bit[4] ^ crc\_bit[3] ^ crc\_bit[1] ^

crc\_bit[0];

new\_bit[4] = crc\_bit[31] ^ crc\_bit[30] ^ crc\_bit[29] ^ crc\_bit[25] ^ crc\_bit[24] ^ crc\_bit[20] ^ crc\_bit[19] ^

crc\_bit[18] ^ crc\_bit[15] ^ crc\_bit[12] ^ crc\_bit[11] ^ crc\_bit[8] ^ crc\_bit[6] ^ crc\_bit[4] ^

crc\_bit[3] ^ crc\_bit[2] ^ crc\_bit[0];

new\_bit[3] = crc\_bit[31] ^ crc\_bit[27] ^ crc\_bit[25] ^ crc\_bit[19] ^ crc\_bit[18] ^ crc\_bit[17] ^ crc\_bit[15] ^

crc\_bit[14] ^ crc\_bit[10] ^ crc\_bit[9] ^ crc\_bit[8] ^ crc\_bit[7] ^ crc\_bit[3] ^ crc\_bit[2] ^

crc\_bit[1];

new\_bit[2] = crc\_bit[31] ^ crc\_bit[30] ^ crc\_bit[26] ^ crc\_bit[24] ^ crc\_bit[18] ^ crc\_bit[17] ^ crc\_bit[16] ^

crc\_bit[14] ^ crc\_bit[13] ^ crc\_bit[9] ^ crc\_bit[8] ^ crc\_bit[7] ^ crc\_bit[6] ^ crc\_bit[2] ^

crc\_bit[1] ^ crc\_bit[0];

new\_bit[1] = crc\_bit[28] ^ crc\_bit[27] ^ crc\_bit[24] ^ crc\_bit[17] ^ crc\_bit[16] ^ crc\_bit[13] ^ crc\_bit[12] ^

crc\_bit[11] ^ crc\_bit[9] ^ crc\_bit[6] ^ crc\_bit[1] ^ crc\_bit[0];

new\_bit[0] = crc\_bit[31] ^ crc\_bit[30] ^ crc\_bit[29] ^ crc\_bit[28] ^ crc\_bit[26] ^ crc\_bit[25] ^ crc\_bit[24] ^

crc\_bit[16] ^ crc\_bit[12] ^ crc\_bit[10] ^ crc\_bit[9] ^ crc\_bit[6] ^ crc\_bit[0];

/\* The new CRC value has been calculated as individual bits in the \*/

/\* new\_bit array. Re-assembled it into a 32 bit value and "clock" it \*/

/\* into the "register". \*/

crc = 0;

for (i = 31; i >= 0; --i) {

crc = crc << 1;

crc |= new\_bit[i];

}

printf("Running CRC value is 0x%08x\n", crc);

}

printf("\n\nThe total number of data words processed was %d\n", data\_count);

printf("The CRC is 0x%08x\n\n", crc);

return 0;

}