Embedded System Design Techniques™

Bootloader Design for MCUs

Session 2: Interface Protocol Design

January 26th, 2016

Jacob Beningo, CSDP

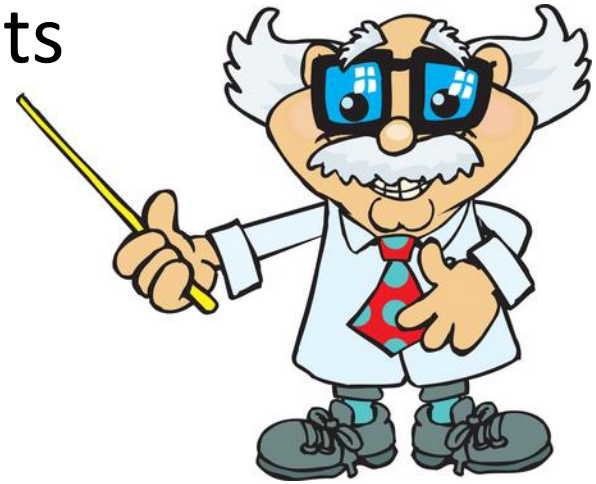
DesignNews

CEC CONTINUING EDUCATION CENTER

Presented by: Digi-Key ELECTRONICS

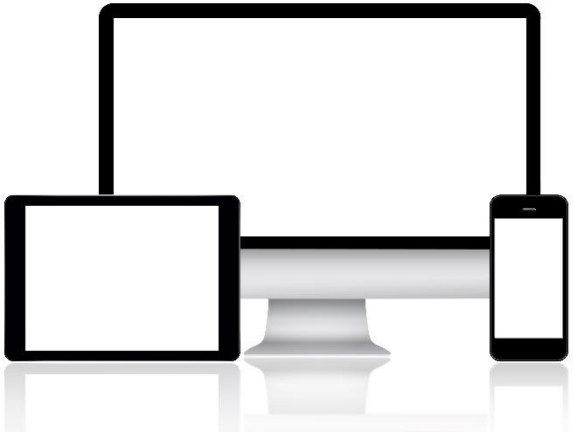
Session Overview

- The Bootloaders Purpose
- Application Storage Formats
- Message Packet Format
- Converting Application Formats



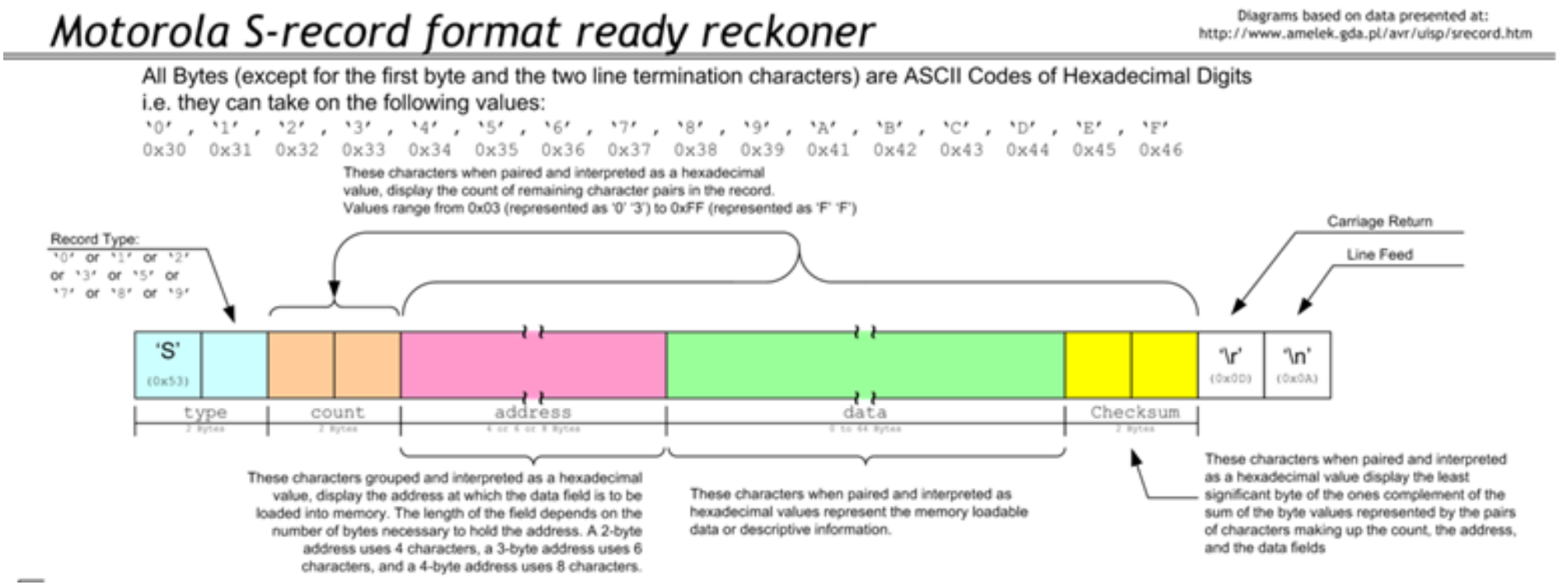
The Bootloaders Purpose

Environment



MCU

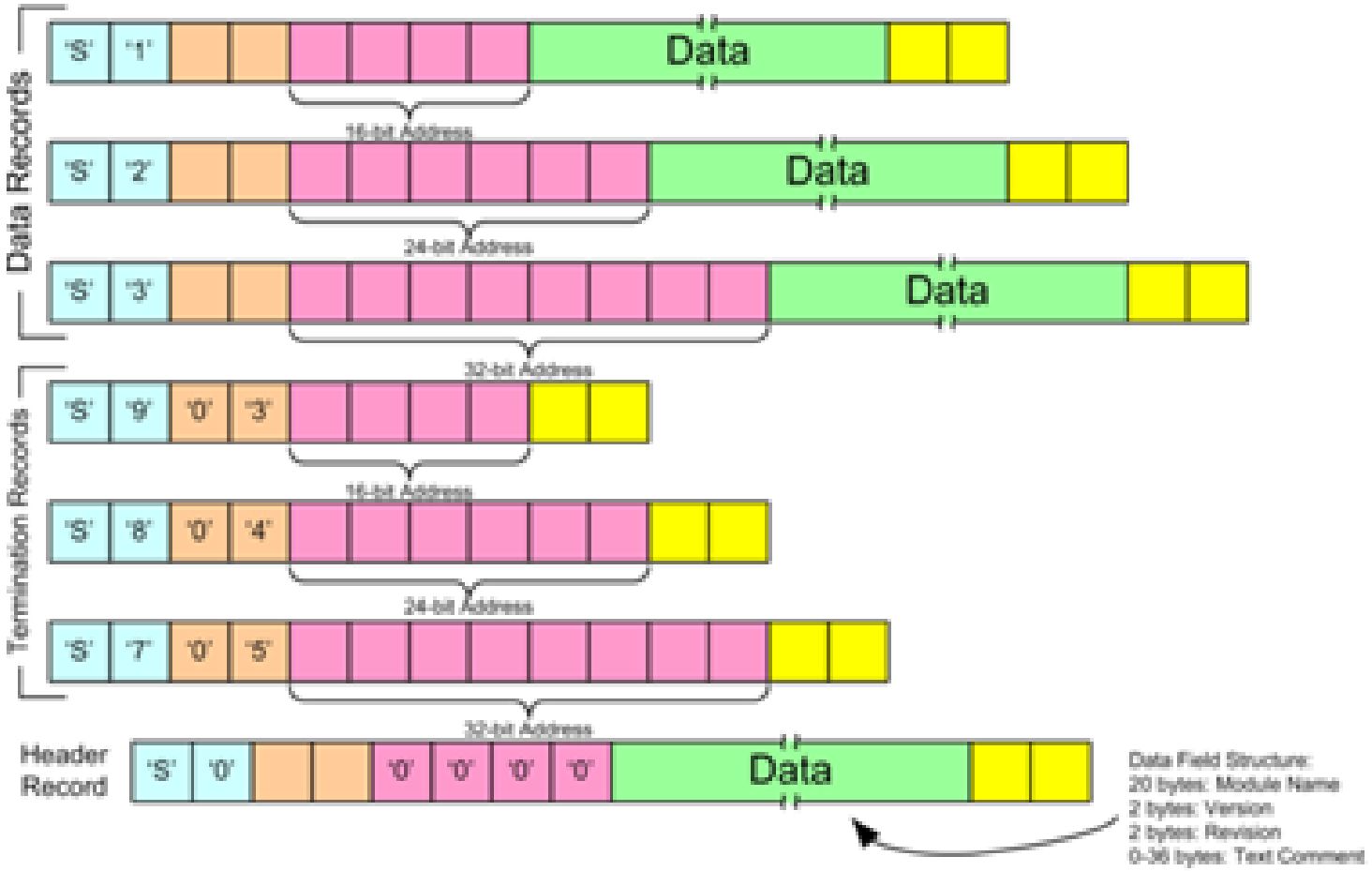
Application Storage Format



https://en.wikipedia.org/wiki/SREC_(file_format)#/media/File:Motorola_SREC_Chart.png

Split in half to fit on slides in a legible fashion

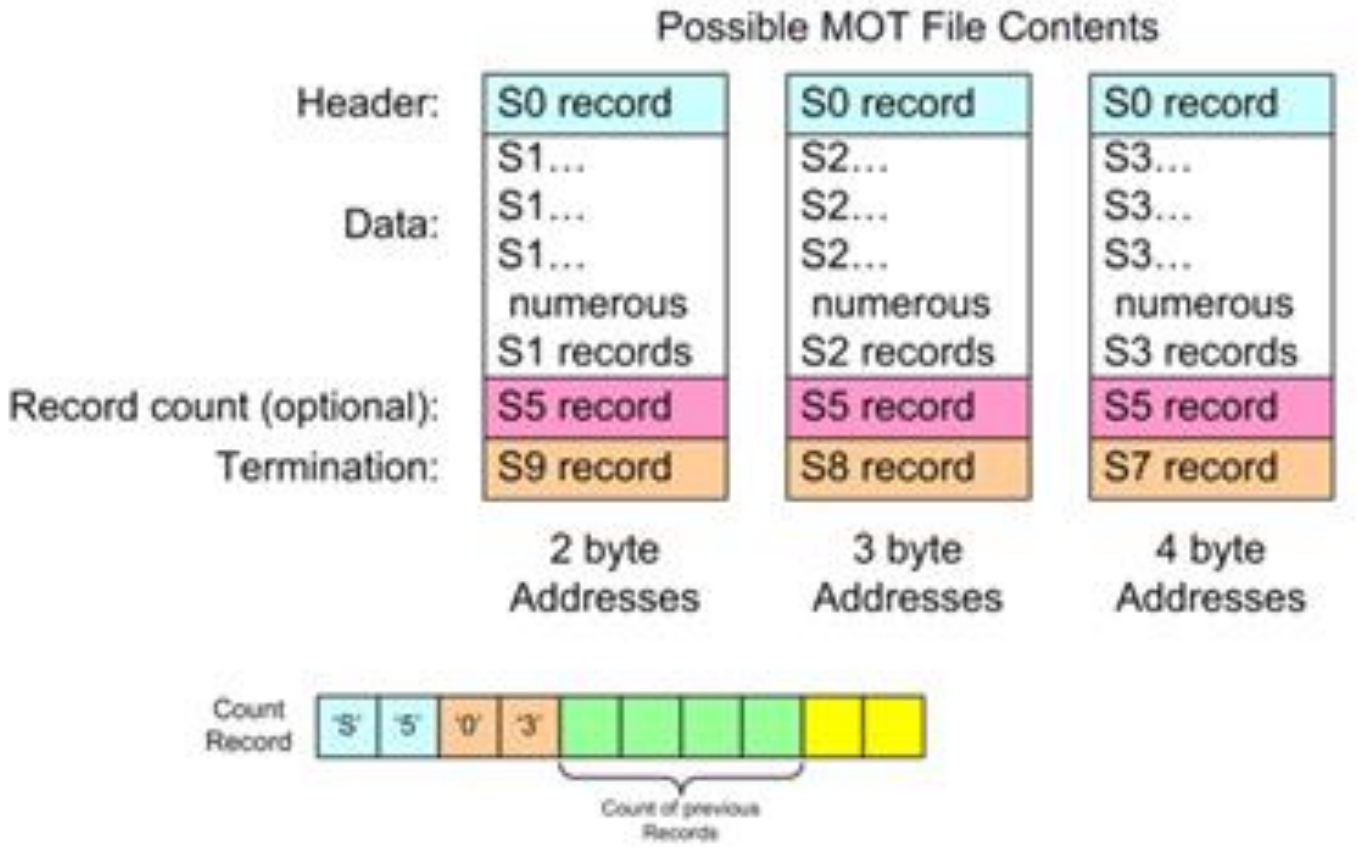
Application Storage Format



https://en.wikipedia.org/wiki/SREC_(file_format)#/media/File:Motorola_SREC_Chart.png

Split in half to fit on slides in a legible fashion

Application Storage Format



https://en.wikipedia.org/wiki/SREC_(file_format)#/media/File:Motorola_SREC_Chart.png

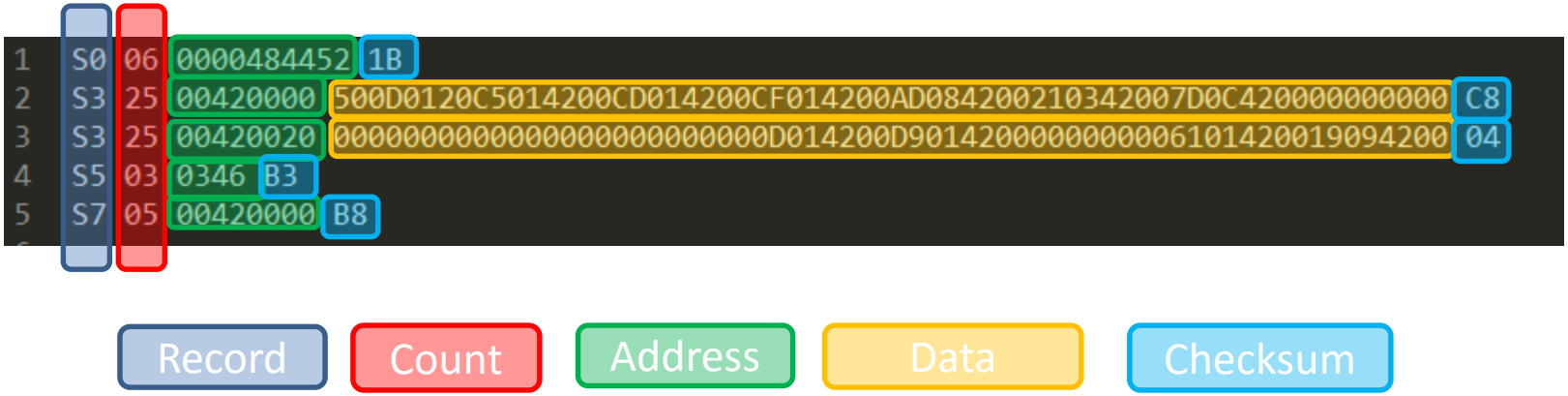
Split in half to fit on slides in a legible fashion

Application Storage Format

1)

```
1  S00600004844521B
2  S3250042000050OD0120C5014200CD014200CF014200AD08420021034200 7D0C420000000000C8
3  S3250042002000000000000000000000000000D014200D90142000000000061014200190942000 4
4  S5030346B3
5  S7050042000 0B8
```

2)



Message Packet Format

- General Message Format

Start Message (8 bits)	OPCODE (8 bits)	Data Length (8 bits)	Data (x bytes)	Checksum (16 bits)

- Checksum
  - Fletcher16 (Approaches error detection of CRC)
  - https://en.wikipedia.org/wiki/Fletcher%27s_checksum

Message Packet Format

```
uint16_t Packet_ChecksumCalculate(uint8_t const * Data, uint16_t Bytes)
{
  uint16_t sum1 = 0xff, sum2 = 0xff;
  uint16_t tlen;

  while(Bytes)
  {
    tlen = Bytes > 20 ? 20 : Bytes;
    Bytes -= tlen;

    do{
      sum2 += sum1 += *Data++;
    } while (--tlen);

    sum1 = (sum1 & 0xff) + (sum1 >> 8);
    sum2 = (sum2 & 0xff) + (sum2 >> 8);
  }

  // Second reduction step to reduce sums to 8 bits
  sum1 = (sum1 & 0xff) + (sum1 >> 8);
  sum2 = (sum2 & 0xff) + (sum2 >> 8);

  return (sum2 << 8 | sum1);
}
```

Message Packet Format

OPCODE	Command	Description
0x30	Bootloader Enter	This command is used to put the system into boot-loader mode.
0x31	Bootloader Exit	Used to exit the bootloader with the intention of entering the application code.
0x32	Device Erase	Erases the application buffer space and prepares for receipt of new application code.
0x33	Device Program	S-Record to program to the application buffer space.
0x34	Device Secure	Secures the flash space from being read and written
0x35	Device Unsecure	Unsecure the flash space for writing and reading.
0x36	Query Device	Used to determine if the system is in bootloader or application mode.

Message Format

```
45 /**
46 * Defines the commands being received by the bootloader.
47 */
48 typedef enum
49 {
50   BOOT_ENABLE = 0x30,        /**< Enter bootloader */
51   BOOT_EXIT = 0x31,          /**< Exit bootloader */
52   ERASE_DEVICE = 0x32,       /**< Erase application area of memory */
53   PROGRAM_DEVICE = 0x33,     /**< Program device with an s-record */
54   QUERY_DEVICE = 0x34,       /**< Set the Slave Tx buffer with the current state */
55   COPY_APPLICATION = 0x35,   /** Copy application from buffer to app space */
56
57   END_OF_COMMANDS       /**< End of command list */
58 }BootCommand_t;
59
60 /**
61 * Defines the responses sent by the bootloader.
62 */
63 typedef enum
64 {
65   IDLE = 0x00,             /**< Bootloader is idle */
66   ERASE_FINISHED = 0x01,   /**< Erase finished */
67   WRITE_FINISHED = 0x02,   /**< Write finished */
68   CHECKSUM_FAIL = 0x03,    /**< Checksum error */
69   BUSY = 0x04,             /**< Bootloader busy */
70   COPY_FINISHED = 0x05,    /**< Copy Completed */
71   RECORD_APP_SPACE_INVALID = 0x10,
72   PACKET_INVALID = 0x20,
73
74   APP_MODE = 0x01,       /**< Application running */
75   BOOT_MODE = 0x03       /**< Bootloader running */
76 }BootResponse_t;
```

Message Packet Format

- Best Practices
  - Use a packet format
  - Use a checksum or CRC
  - Track record packet numbers
  - Use ACK and NAK for each packet
  - Create specific error codes
  - Don’t assume that errors rarely happen
  - Use 32 bit addressing as a default
  - Include response timeout

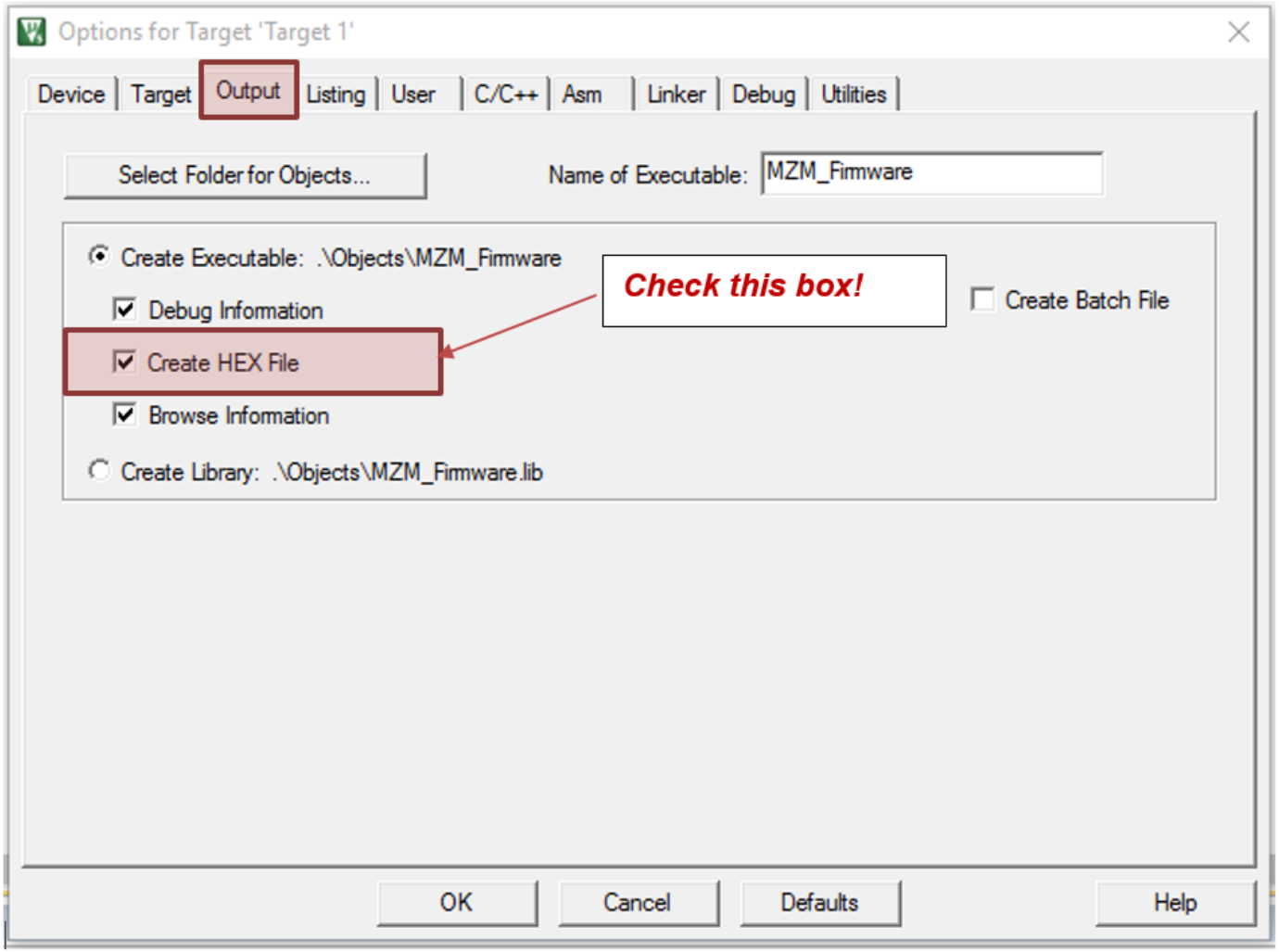
Converting Application Formats

Some IDEs only put out one record format!

Useful format converter

- Hex2bin (http://hex2bin.sourceforge.net/)
- Bin2Srec (http://www.s-record.com/)
- Many other tools

Converting Application Formats



Converting Application Formats

- Hex2bin
  - hex2bin filename

```
F:\>hex2bin MZM_Firmware.hex
hex2bin v2.1, Copyright (C) 2015 Jacques Pelletier & contributors

Lowest address:   4194304
Highest address:  4219631
Starting address: 4194304
Max Length:       25328

Binary file start = 00400000
Records start     = 00400000
Highest address   = 004062EF
Pad Byte          = FF

F:\>
```

Converting Application Formats

- Bin2Srec
  - Bin2srec –a 4 –o 0x420000 input_filename > output_filename

```
F:\>bin2srec -a 4 MZM_Firmware.bin > MZM_Firmware.srec

BIN2SREC 1.46 - Convert binary to Motorola S-Record file.
Copyright (c) 2000-2015 Ant Goffart - http://www.s-record.com/

Input binary file: MZM_Firmware.bin
Begin address   = 0h
End address     = 62EFh
Address offset  = 0h
Maximum address = 62EFh
Address bytes   = 4
Processing complete

F:\>bin2srec -a 4 MZM_Firmware.bin > MZM_Firmware.srec
```

Additional Resources

- Download Course Material for
  - Updated C Doxygen Templates (Sept 2015)
  - Example source code
  - Bootloader White Paper
  - Templates
- Microcontroller API Standard
- EDN Embedded Basics Articles
- Embedded Bytes Newsletter

From www.beningo.com under

- Blog and Articles > Software Techniques > CEC Bootloader Design for MCUs

DesignNews

Presented by:

CEC CONTINUING EDUCATION CENTER

Digi-Key ELECTRONICS

Jacob Beningo
Principal Consultant

P.O. Box 400
Linden, Michigan 48451

www.beningo.com

E : jacob@beningo.com
T : 810-844-1522
: Jacob_Beningo
: Beningo Engineering
: JacobBeningo
EDN : Embedded Basics

IEEE computer society Certified Software Development Professional

ARM Connected Community

Newsletters

- Embedded Bytes

http://bit.ly/1BAHYXm

Training

- MicroPython
- Bootloaders
- Low Power Design
- Resume Workshop
- Embedded Techniques

Presented by:

DesignNews

CEC CONTINUING EDUCATION CENTER

Digi-Key ELECTRONICS