

Everest-Frequency-Synthesizer-Demo

Getting Started

Project: Everest-Frequency-Synthesizer-Demo		created:	S. Rieche	Date	2018-02-19
Getting Started	edited:	S. Rieche	Date:	2019-01-28	
		approved:		Date:	
Filename: Everest-Freq_Synths-DemoGetting_Started_1p2.docx					
Arrow Central Europe GmbH		Version:	1.2	Page 1 of 2	20

Contents

1.	Revis	ion History5	;
1.1	Rev	ision 1.15	;
1.2	Rev	ision 1.05	5
2.	Getti	ng Started 6	5
2.1	Pre	requisites6	5
2.2	Han	dling the Board7	7
2.3	Boa 2.3.1	rd-Setup Revision PROTO	7
	2.3.2	Toggle -Switch S5 – SC SPI-Flash enable7	7
	2.3.3	DIP-Switch S8 – FMC Voltage Selector7	7
	2.3.4	Toggle -Switch S9 – VDDAUX2 & VDDAUX5 Voltage	7
2.4	Boa 2.4.1	rd-Setup Revision A and B	3
	2.4.2	Toggle -Switch S5 – SC SPI-Flash enable 8	}
	2.4.3	DIP-Switch S8 – FMC Voltage Selector 8	3
	2.4.4	Toggle -Switch S9 – VDDAUX2 & VDDAUX5 Voltage 8	3
2.5	Pov	vering up the Board)
3.	Demo	o Design)
3.1	Pre	requisites10)
3.2	Des	ign Implementation10)
3.3	Ger	neration of the ZL30265 and ZL30722 eeprom content11	ł
3.4	Rur	ning the Design16	5

Figures

Figure 1: Everest Board	9
Figure 2: Design Implementation	11
Figure 3: ZL30267 Evaluation Board Software, Version 1.2 - Save Configuration	12
Figure 4: ZL30267 Evaluation Board Software, Version 1.6 - Create EEPROM Image File	12
Figure 5: ZL30722 configuration tool (Host GUI - Rev 2.3.1)	14
Figure 6: ZL30722 configuration	15
Figure 7: ZL30722 - Create EEPROM Image File	15
Figure 8: SoftConsole v5.1 workspace launcher	16
Figure 9: SoftConsole v5.1 - starting the debug session	17
Figure 10: SoftConsole v5.1 - running the design	17
Figure 11: terminal output	18
Figure 12: ZL30265 eeprom content	19
Figure 13: Frequency measurement	20

Tables

Table 1: Software / IP Requirements	10
Table 2: Hardware Design Clock Frequencies	10

1. Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 1.2

The document was updated for Libero SoC v12.0.

1.2 Revision 1.1

The document was updated for Libero SoC PolarFire v2.2.

1.3 Revision 1.0

Revision 1.0 is the first publication of this document.

2. Getting Started

This demo design can be used to verify the functionality of Microsemi's frequency synthesizer ZL30265. Beside that two clock outputs of the Microsemi ZL30722 system synchronizer clock generator are monitored. Both circuits are mounted on the Everest DEV Board. The demo design is based on the Everest-CortexM1-Demo design that implements a Cortex M1 soft processor subsystem with GPIO's and UART functionality. Internal SRAM blocks are used for both, program and data memory. One UART terminal is operated through USB connector J9. In addition, two SPI cores are used to configure the circuits by firmware. For each of the nine clock inputs a frequency counter is instantiated. An AHB slave is used to start the frequency counters and read out their counter values. For debugging purpose three clock signals are divided by 100 and could be observed on the PMOD pins 7, 8 and 9. The ZL30722 SPI signals are routed to the PMOD pins 1 to 4.

The application prints a menu on the terminal with that the status, global, APPL, input and output registers of the ZL30265 could be read out. Further the eeprom content of both circuits could be read and programmed with precompiled eeprom images. A frequency measurement is started by hitting the 'm' key. LEDs 1 to 3 are toggled by a software counter in the main loop, that could be reseted by pressing one of the four push buttons. LED 4 is on when a frequency measurement is running (one second).

2.1 Prerequisites

Item	Quantity
Everest DEV Board	1
12 V / 5 A wall-mounted power adapter	1
USB 2.0 A male to mini-USB B cable for	1
UART / Programming interface to PC	
Free one-year Libero Silver software license	1

For the Everest Cortex M1 Demo the following is needed:

Note 1: The Everest DEV Board offers an on-board FlashPro5 programmer, which can be used to program and debug with Identify, SmartDebug and embedded application software using SoftConsole.

Note 2: The descripted design is suitable for Everest Dev Board Rev. PROTO, A and B.

2.2 Handling the Board

Pay attention to the following points while handling or operating the board:

Handle the board with electrostatic discharge (ESD) precautions to avoid damage.

For information about ESD precautions see

https://www.microsemi.com/documentportal/doc_view/126483-esd-appnote.

2.3 Board-Setup Revision PROTO

2.3.1 Toggle-Switch S1 – PCle

Warning: S1-1 and S1-2 must not be at position on at the same time!

SWITCH ON	PCIe LANES
S1-1	x1
S1-2	x4

2.3.2 Toggle -Switch S5 – SC SPI-Flash enable

Warning: S5-1 and S5-2 must not be at position on at the same time!

SWITCH ON	SC SPI-FLASH
S5-1	ENABLE
S5-2	DISABLE

2.3.3 DIP-Switch S8 – FMC Voltage Selector

Warning: S8-1 to S8-4 must not be at position on at the same time!

SWITCH ON	FMC VOLTAGE
S8-1	3.3 V
S8-2	2.5 V
S8-3	1.8 V
S8-4	undefined (not connected)

2.3.4 Toggle -Switch S9 – VDDAUX2 & VDDAUX5 Voltage

Warning: S9-1 and S9-2 must not be at position on at the same time!

SWITCH ON	VDDAUX2 & VDDAUX5
S9-1	2.5 V
S9-2	FMC voltage

2.4 Board-Setup Revision A and B

2.4.1 Toggle-Switch S1 – PCle

SWITCH	PCIe LANES
S1-1 (marking)	x4
S1-2	x1

2.4.2 Toggle -Switch S5 – SC SPI-Flash enable

SWITCH	SC SPI-FLASH
S5-1 (marking)	DISABLE
S5-2	ENABLE

2.4.3 DIP-Switch S8 – FMC Voltage Selector

SWITCH	FMC VOLTAGE
S8-1 off, S8-2 off	1.8 V
S8-1 on, S8-2 off	2.5 V
S8-1 off, S8-2 on	undefined (not recommended)
S8-1 on, S8-2 on	3.3 V

2.4.4 Toggle -Switch S9 – VDDAUX2 & VDDAUX5 Voltage

SWITCH	VDDAUX2 & VDDAUX5
S9-1 (marking)	2.5 V
S9-2	FMC voltage





Figure 1: Everest Board

2.5 Powering up the Board

The Everest DEV Board is powered up using either the 12 V DC jack or the PCIe connector. For programming connect it although with your computer using USB mini B connector J9.

3. Demo Design

3.1 Prerequisites

Table 1: Software / IP Requirements

Software	Version
Libero SoC PolarFire	V12.0
Synplify Pro	L2017.09M-SP1-1
FlashPro PolarFire	V2.0
IP	
CortexM1	3.0.100
PF_SRAM_ABHL_AXI	1.1.127
PF_INIT_MONITOR	2.0.103
CoreAHBLite	5.3.101
CoreAHBTOAPB3	3.1.100
CoreAPB3	4.1.100
PF_OSC	1.0.102
PF_CCC	1.0.115
PF_XCVR_REF_CLK	1.0.103
CoreUARTapb	5.6.102
CoreGPIO	3.2.102
CORESPI	5.2.104

Before you start you have to make sure, that all cores are downloaded to your local vault.

3.2 Design Implementation

The following table lists the clock frequencies used in the design.

Table 2: Hardware Design Clock Frequencies

Clock	Frequency (MHz)
PF_OSC	160
PF_CCC OUT0_FABCLK	80
HCLK / PCLK	80



Figure 2: Design Implementation

The design is already fully implemented and ready to be programmed on the Everest Board. The board has to be connected with the power supply and to the PC with the USB cable. All drivers have to be installed (which should happen automatically when plugged in the first time) To program the design, there are two possibilities:

- Programming via Libero PolarFire SoC: Programming is started with the "Run PROGRAM Action" Button in the Design Flow Pane
- Programming via FlashPro Software: There is a STAPL-File ("<Design Directory>\designer\CortexM1_Subsystem\export\CortexM1_Subsystem.stp") which can be programmed with the FlashPro Software. A new FlashPro project has to be generated and the programming file loaded into.

3.3 Generation of the ZL30265 and ZL30722 eeprom content

The configuration of the ZL30265 is done with the graphical tool "ZL30267 Evaluation Board Software". Due to a software bug, two versions were used. Version 1.2 to create the config file (*.mfg) and in a second step version 1.6 to load the config file and export a text file (*.txt) with the eeprom content. Figure 3 shows the configuration and how to save it in a config file.

Arrow Central Europe GmbH

User Guide

ZL30265 - Microsemi		- 0
File Connect Refresh Reset Registers	GPIO Log Documentation About	
Load Configuration		
Save Configuration		
Exit		
XA		Bank A Source: APLL1 Int Divider V 600,0000000 MHz
Doubler Invert	Working Offline	Format Divisor Output Frequency
Mode : Clask Sizes		OC1 LVDS v 12 50,000000 MHz S
Clock Signal	APLL1	
XA Frequency (MHz):	Enable APLL Enable Frac Divider	
125,0000000 Frac	Doubler Align Output Dividers Status: Locked	Bank B Source: APLL1 Int Divider V 600,0000000 MHz
125 000000 MHz	Selected Source: XA VCO Frequency (MHz):	Format Divisor Output Frequency
120,000000 1112	Feedback Signal: FB Divider v 3900,0000000 Frac	OC3 LVDS ~ 8 75,000000 MHz S
	Source Selection: Reg Controlled	
IC1	Source: X4 S00 000000 6.5 X	
Enable Invert		Bank C Source: APLL1 Int Divider V 600,0000000 MHz
IC1 Frequency (MHz):	Frequency After Frac Divider (MHz):	Format Divisor Output Frequency
50,0000000 Frac	Undefined Frac	OC4 LVDS V 6 100,0000000 MHz S
Frequency After Divider:	Enable Bypass Path 1	
50,000000 MHz		
	APLE2	Bank D Source: APLL2 Int Divider V 625,0000000 MHz
IC2	Artiz	Format Divisor Output Frequency
Enable Invert	Doubler Alian Output Dividers Status Locked	OC6 LVDS V 5 125.000000 MHz S
IC2 Frequency (MHz):	Selected Source: XA	
50,0000000 Frac	VCO Frequency (MHz):	
Frequency After Divider:	Feedback Signal: FB Divider V 3/50,0000000 Frac	Bank E Source: APLL2 Int Divider V 625,0000000 MHz
50,0000000 MHz	Source Selection: Reg Controlled Frequency After Int Divider (MHz):	Format Divisor Output Frequency
	Source: XA ~ 625,000000 6 ~	OC8 LVDS V 4 156,2500000 MHz S
IC3	Frequency After Frac Divider (MHz):	
Enable Invert	Undefined Frac	
IC3 Frequency (MHz):	Enable Bypass Path 2	Bank F Source: APLL2 Int Divider V 625.0000000 MHz
50,0000000 Frac		Format Divisor Output Frequency
Frequency After Divider:		OC9 LVDS V 4 156,2500000 MHz S
50,0000000 MHz	🛇 Microsemi.	

Figure 3: ZL30267 Evaluation Board Software, Version 1.2 - Save Configuration

After opening Version 1.6 of the ZL30267 Evaluation Board Software and loading the config file, the eeprom image could be created by clicking "EEPROM -> Create EEPROM Image File", as shown in Figure 4.



Figure 4: ZL30267 Evaluation Board Software, Version 1.6 - Create EEPROM Image File

In the eeprom image text file every byte has its own line. The following listing shows the first

lines of the text file.

```
; EEPROM Image File
                 : ZL30265
; Device Id
; Device Revision : 1
; GUI Version
                 : 1.6
; File Generation Date : Donnerstag, 30. November 2017 13:00:06
; File Format
                : ASCII
; Config File (0): C:\Users\Rieche\Desktop\ZL30265-50-75-100-125-156 25.mfg
    XA INPUT FREQUENCY HZ = 125000000 * (1) / (1)
IC1 INPUT FREQUENCY HZ = 50000000 * (1) / (1)
;
                                   * (1) / (1)
;
    IC2 INPUT FREQUENCY HZ = 50000000 * (1) / (1)
;
                                   * (1) / (1)
   IC3 INPUT FREQUENCY HZ = 50000000
;
    APLL1 VCO FREQUENCY HZ = 3900000000 * (1) / (1)
;
                                   * (1) / (1)
    FDIV1 OUTPUT FREQUENCY HZ = UNDEFINED
;
   APLL2 VCO FREQUENCY HZ = 3750000000 * (1) / (1)
;
   FDIV2 OUTPUT FREQUENCY HZ = UNDEFINED * (1) / (1)
;
   Reg Write Commands: 44, Wait Commands: 1
;
; EEPROM File Statistics
    Preamble Bytes
                         : 26
;
    Number of Bytes in Branch0 : 105
;
    Checksum Byte : 1
Total Bytes Needed : 132/2036
;
;
; Checksum: modulo 256 add of all bytes thru checksum should be 0.
; Address range 07F4 to 07FF is reserved for factory test and should not be modified.
; This address range is omitted from the EEPROM data image to prevent it from being
overwritten.
80
21
00
00
84
44
00
0F
1A
00
00
00
00
00
00
00
...
```

This data could be used to create a c-header file (*.h) like in the following listing, that could be include in the firmware project.

User Guide

<pre>#ifndef ZL30265_IMAGE_H_</pre>				
#define ZL30265_IMAGE_H_				
#include <stdint.h></stdint.h>				
<pre>#define ZL30265_IMAGE_SIZ</pre>	E (sizeof(Z)	130265_image)	/ sizeof (ui	nt8_t))
const uint8 t ZL30265 ima	ge[] = {			
/* 0x80, 0x2	l, 0x00, 0x00,	0x84, 0x44,	0x00, 0x0F,	
/* 8 */ 0x1A, 0x0), 0x00, 0x00,	0x00, 0x00,	0x00, 0x00,	
/* 16 */ 0x00, 0x0), 0x00, 0x00,	0x00, 0x00,	0x00, 0x00,	
/* 24 */ 0xFF, 0xF	F, 0x04, 0x24,	0x0B, 0x40,	0x02, 0x01,	
/* 32 */ 0x02, 0x0	3, 0x40, 0x05,	0x01, 0xAD,	0x01, 0x00,	
/* 40 */ 0x08, 0x2	A, 0x01, 0x01,	0x05, 0x41,	0x06, 0x06,	
/* 48 */ 0x00, 0x6	6, 0x66, 0x66,	0x3E, 0x00,	0x05, 0x01,	
/* 56 */ 0x10, 0x0	l, 0x01, 0x14,	0x09, 0x01,	0x20, 0xC7,	
/* 64 */ 0x01, 0x2	2, 0x5F, 0x41,	0x26, 0x01,	0x98, 0x90,	
/* 72 */ 0x41, 0x8	6, 0x05, 0x00,	0x00, 0x00,	0x00, 0x3C,	
/* 80 */ 0x00, 0x0	L, 0x94, 0x09,	0x01, 0xA7,	0x90, 0x42,	
/* 88 */ 0x00, 0x0	l, 0x0B, 0x01,	0x42, 0x20,	0x01, 0x07,	
/* 96 */ 0x01, 0x4	2, 0x30, 0x01,	0x05, 0x01,	0x42, 0x50,	
/* 104 */ 0x01, 0x0	4, 0x01, 0x42,	0x70, 0x01,	0x03, 0x01,	
/* 112 */ Ux42, Ux8), 0x01, 0x03,	0x01, 0x04,	0x30, 0x0C,	
/* 120 */ 0x04, 0x3), 0x00, 0x04,	0x35, 0x0C,	0x04, 0x35,	
/* 128 */ 0x00, 0xD	3, UXFF, UXFI	0 55 0 55	0 55	
/* 32 byte boundary	IIII */, UXFF	, UXFF, UXFF,	UXF'F'	
};				
<pre>#endif /* ZL30265_IMAGE_H</pre>	*/			

The creation of the ZL30722 eeprom image header file is similar to this process, but without the need to use two different software version of the tool (Figure 5, Host GUI – Rev 2.3.1).



Figure 5: ZL30722 configuration tool (Host GUI - Rev 2.3.1)

Figure 6 and Figure 7 are showing the actual configuration of the ZL30722 and how to create an eeprom image file.

🗢 ZL30722 Host GUI - Rev 2.3.1			– 🗆 X
File Connection Tools			
Part Number: ZL30722 rev AB	Device Mode: Jitter Attenuation (DPLL+APLL) V	Working Offline	Polling Rate: 1 Seconds 🗸 🗸
DPLL Master Clock Reference Type: TCXO ✓ Frequency (MHz): 125.000000 Freq Adjust (ppm): 0.000	DPLL Master Clock Mode: Common PFD Freq (MHz): 0.231481 Bandwidth (Hz): 1.7 State: Freerun	APLL State: Locked VCO Frequency (MHz) 3750.000000 HSDIV1 Frequency (MHz)	Outputs OC1 Config Status: Enabled Format: CML (800mV) ~ Frequency (MHz) (0=Unused) 125.00000 Frac
Inputs IC1 Enable Pins Invert Frequency (MHz) (0=Unused) O.000000 Frac Config Status IC2 Enable Pins Invert	Selected Reference: Automatic Frequency Offset (ppm): 0.00000000 Phase (ne): 0.000 DPLL Show NCO	750.0000000 HSDIV2 Frequency (MHz) 0.0000000 APLL	OC2 Config Status: Enabled Fomat: CML (800mV) ~ Frequency (MHz) (0=Unused) 125,000000 Frac
Frequency (MHz) (0=Unused) 0.00000 Frac Config Status IC3 Invet Frequency (MHz) (0=Unused) 0.00000 0.000000 Frac Config Status	Quick Links GPIO Config Phase Adjust Phase Measure	Power Estimate Start Log	OC3 Config Status: Enabled Format: CML (800mV) ↓ Frequency (MH2) (0=Unused) 0.000000 Frac
Host GUI Version: (2.3.1) :: (08091700)			.:

Figure 6: ZL30722 configuration

C ZL30722 Host GUI - Rev 2.3.1		
File Connection Tools		
Load Configuration 🔸		
Save Configuration 🔸		
EEPROM >	Create EEPROM Image File	
Register Dump File 🔹 🕨	Write/Erase Image To EEPROM	
Exit	Dump DUT EEPROM to File	
Reference Type: TCXO	Verify DUT EEPROM Against File le: Automatic 🗸	
Frequency (MHz): 125,0000	0 Clock Common PFD Freg (MHz): 0,231481	
Freq Adjust (ppm): 0,000	Bandwidth (Hz): 17	
	State: Freerun	
	Selected Reference: Automatic ~	
Inputs	Frequency Offset (ppm): 0,00000000	
IC1	Phase (ns): 0,000	
Enable Pins Invert Frequency (MHz) (0=Unused)	DPLL Show NCO	
0,000000 Frac		

Figure 7: ZL30722 - Create EEPROM Image File

The c-header file has to be create in the same way as explained above.

3.4 Running the Design

In Order to run the design, the CortexM1-Processor has to be loaded with the firmware. To do so, load the provided SoftConsole Workspace.

Note: Debugging of Cortex-M1 applications is currently supported only in SoftConsole v5.1. A future release of SoftConsole will support this feature.

SC Workspace Launcher	\times
Select a workspace	
Microsemi SoftConsole v5.1 stores your projects in a folder called a workspace.	
Choose a workspace folder to use for this session.	
	_
Workspace: ROJEKTE\Arrow\FPGA-Design\PolarFire_CortexM1_Freq_Synths\softconsole Browse	
Use this as the default and do not ask again	
	_
OK Cancel	
Mierocomi	
Iviicroserni.	

Figure 8: SoftConsole v5.1 workspace launcher

User Guide

SC C/C++ - PolarFire_CortexM1_Freq_Synths/m File Edit Source Refactor Navigate Sea	ain.c - Microsemi SoftConsole v5.1 Irch Project Run Window Help					- a ×
- 🛛 🛍 🗁 🕲 - 🗞 - 🛍 🖾 👘	× 2 · · · · · · · · ·	· • • • • • • • • • • • • • • • • • • •	• 🍦 • 👳 🖕 • 🗅 •			Quick Access
bright Explorer 12		<pre>mass 11 @ freg.yethic d</pre>	10, LANE_0, 6_DADAS_DIEK 6_DADAS_DIEK 11, LATE_0, 12_DADAS_DIEK 12_DADAS_DIEK 13_DADAS_DIEK 10((42)AATO, 42_DIDAGAS), break; 10((42)AATO, 42_DIDAGAS, 0010, 0020, 10((42)AATO, 42_DIDAGAS, 0010, 0010, 10((42)AATO, 42_DIDAGAS, 0010, 10((42)AATO, 42_DIDAGAS, 0010, 10((42)AATO, 42_DIDAGAS	"satus"); break; "Godsi"); break; "Jodsi"); break; "Jour clock"; break; "Input clock"; break;	~	<pre>Section 1: @ Mate Target B Text isit @ Dec platform.h Dec platform.h De</pre>
				Writable	Smart Insert 1	105 : 58

Figure 9: SoftConsole v5.1 - starting the debug session

A debug configuration is provided to download the firmware to the CortexM1 processor and start the application.

SC Debug - PolarFire_ContexM1_Freq_Synths/main.c - Microsemi SoftConsole v5.1						- o ×
	A + 📕 > 4 + 5 + 5 - 5 + 5 +				Quick Access	C/C++ the Debug
the Debug 22		(*)= Variables 22 🗣 Breaknoints 1997	Registers 🗟 Peripherals 🛋 Modules		1	
v (II) Peterferic, Controll J. Frag. Synchol Boboy (IOB OpenOCD Debogging) v (II) Peterferic, Controll J. Frag. Synchol II (Boognood) v (III) Peterferic (Control III) Peterferic (III) Peterferic (IIII) Peterferic (III) Pet		Name 60- top 00- top 00- gunt0 00- gun0	Type uRL1 uRL2 URT_Instance_1 URT_Instance_1 spl_instance_1 spl_instance_1 freq_synthe_1 freq_synthe_3		Value 0 ∿0* 5667 1→ 1→ 1→ 1→ 1→ 1→	
a main c 🕴 🖻 free surths c				Se Outline 22	s m A.	> □ □ □ ₩ • ₩ • ₩
<pre>content = termination = t</pre>				bh uguletorm. bh uguletorm. driver:Corelike drin)/core.gpic.h Taph/core.gpic.h Taph/core.gpi.h Inter_mic.ch 1(_dg.h Sonteem1_cdg.h Sonteem1_cdg.h (_dg.h) Sonteem1_cdg.h (_gg.h) Sonteem1_cdg.h (gg.h) Sonteem1_cdg.h (gg.h) Sonteem1_cdg.h (gg.h) Sonteem1_cd	×_(1', gpio_instance_(1') ;vo
📴 Console 🕸 🧟 Tasks 🕵 Problems 💽 Executables 🚦 Memory				= 3	K 🔆 🔒 🔒 🗊 🖓 🥵	🛃 📮 🕇 🕇 🗖 🗖
<pre>rearry_content_ire_yeths topuggent generation percect presequences of a start of a</pre>						Î
<			Writable	Smart Insert 68 : 1		>

Figure 10: SoftConsole v5.1 - running the design

Arrow Central Europe GmbH

User Guide

Putty COM15 - Putty	_	×
c control (interactive register write)		
s read status registers		
g read global registers		
a read APPL registers		
o read output registers		
i read input registers		
r read ZL30265 eeprom		
w write ZL30265 eeprom		
0 write ZL30722 eeprom		
l read ZL30722 eeprom		
m measure counters		
		\sim

Figure 11: terminal output

The contend of ZL30265 eeprom could be read out by pressing the 'r' key.

User Guide

🖉 COM15 - PuTTY		-	×
			<u>^</u>
freq_synths_read_e	eeprom		
addr: 00000000 80 2	21 00 00 84 44 00 OF		
addr: 00000000 IA (
addr: 00000010 00 0			
addr: 00000018 FF F	P 04 24 0B 40 02 01		
addr: 00000020 02 0	23 40 05 01 AD 01 00		
addr: 00000020 00 2	56 66 66 3F 00 05 01		
addr: 00000038 10 0	11 01 14 09 01 20 C7		
addr: 00000040 01 2	22 5F 41 26 01 98 90		
addr: 00000048 41 8	36 05 00 00 00 00 3C		
addr: 00000050 00 0	01 94 09 01 A7 90 42		
addr: 00000058 00 0	01 0B 01 42 20 01 07		
addr: 00000060 01 4	42 30 01 05 01 42 50		
addr: 00000068 01 0	04 01 42 70 01 03 01		
addr: 00000070 42 8	30 01 03 01 04 30 OC		
addr: 00000078 04 3	80 00 04 35 0C 04 35		
addr: 00000080 00 I	D3 FF F1 FF FF FF FF		
addr: 00000088 53 7	74 61 74 75 73 00 00		
addr: 00000090 47 6	5C 6F 62 61 6C 00 00		
addr: 00000098 41 5	50 4C 4C 00 00 00 00		
addr: 000000A0 75 7	74 20 63 6C 6F 63 6B		
addr: 000000A8 00 0	00 00 49 6E 70 75		
addr: 000000B0 74 2	20 63 6C 6F 63 6B 00		
addr: 000000B8 0A 0	DD 09 63 20 63 6F 6E		
addr: 00000000 74 7	72 6F 6C 20 28 69 6E		
addr: 00000008 /4 8	0 72 65 67 60 72 74		
addr: 00000000 65 2	72 20 77 72 69 74 65		
addr: 00000008 85 1	12 20 11 12 09 14 03		
addr: 000000E8 61 6	54 20 73 74 61 74 75		
addr: 000000E0 73 2	20 72 65 67 69 73 74		
addr: 000000F8 65 7	72 73 0A 0D 09 67 20		
			V

Figure 12: ZL30265 eeprom content

By hitting the 'm' key a frequency measurement is started. It takes one second. The first line simply shows the content of the counters control register. Lines two to ten are showing the counter values. The left column is the hex and the right the decimal representation.

Arrow Central Europe GmbH

User Guide

				_
Rev COM15 - PuTTY		_		×
<u> </u>				
a contro) (interpative register write)			\sim
c contro	Ji (interactive register write)			
s lead a	rlobal registers			
y read yrobai registers				
a leau AFFL legisters				
i read input registers				
r read 7130265 geprom				
I leau a	ZI30265 ceprom			
0 write	ZL30Z03 CEPTOM ZL30Z22 ceptom			
1 read 2	Z130722 eeprom			
m maaguu	re coupters			
in measu	te counters			
read counters:				
80000000	2147483648			
0304EAA9	0050653865			
0304EAA9	0050653865			
04875FFE	0075980798			
0609D552	0101307730			
078C4AA7	0126634663			
096F5D51	0158293329			
096F5D52	0158293330			
078C4D45	0126635333			
078C4D44	0126635332			
				~

Figure 13: Frequency measurement