

ZipWire Bit-pump Version 4.3 Channel Unit Version 6.2 Release Notes

Table of Contents

1	INTRODUCTION	2
2	BIT-PUMP CODE	2
2.1	BUG FIXES	2
2.1.1	<i>Un-initialized Pointer in _InitTxGain().....</i>	<i>2</i>
2.2	ENHANCEMENTS.....	3
2.2.1	<i>Replace SET_WORD with BP_WRITE_REG in _Aagc() Function.....</i>	<i>3</i>
2.2.2	<i>Increase DC Offset Meter Interval</i>	<i>3</i>
3	APPLICATION AND CHANNEL UNIT CODE	5
3.1	BUG FIXES	5
3.1.1	<i>_CU_SET_MFRAME API Does Not Process Zero Correctly</i>	<i>5</i>
3.1.2	<i>LEDs not modified correctly when _NO_OF_LOOPS is less than 3.....</i>	<i>5</i>
3.1.3	<i>T1/E1 Framer Unframed Mode API fix.</i>	<i>6</i>
3.2	ENHANCEMENTS.....	6

1 Introduction

The ZipWire Bit-pump Version 4.3 Channel Unit Version 6.2 release includes changes to the 3 software components: Bit-pump, Channel Unit, and Application Code. The release notes discuss the bit-pump code in Section 2.1; the Channel Unit and Application Code in Section 3.1. This separation allows customers to easily understand the changes and incorporate them into their system.

Verified problems with the release are identified in this section. These changes are incorporated into this software release. Change bars on the margin can identify any changes from one revision to the next of this document.

2 Bit-pump Code

2.1 Bug Fixes

2.1.1 Un-initialized Pointer in _InitTxGain()

The INIT_TX_GAIN macro was replaced by the Init_Tx_Gain() function. The pointer to the bitpump address space, bp_mode_ptr, is declared but it is not initialized. This causes the transmit gain to be un-initialized.

Modify UTIL.C - Starting at Line 810:

```
void _InitTxGain (BP_U_8BIT no)
{
    DECLARE_MODE_PTR;
    BP_S_8BIT cal_tx_gain;

    /*Addition starts*/
    INIT_BP_MODE_PTR;
    /*Addition ends*/

    /* Read Calibrated Tx Gain */
    cal_tx_gain = BP_READ_BIT(bp_mode_ptr, tx_calibrate, tx_gain);
```

2.2 Enhancements

2.2.1 Replace SET_WORD with BP_WRITE_REG in _Aagc() Function

The Microtec 68K compiler gave a warning message when compiling the function _Aagc(). The warning occurred because 2 SET_WORD macros used byte variables instead of double-byte variables as the fourth parameter. The software worked correctly, but it was changed to eliminate the warning, make the software more clear, and to save ROM.

Modify SUUTIL.C - Starting at Line 1585:

Replace:

```
SET_WORD(bp_ptr, far_end_high_alarm_th_low, far_end_high_alarm_th_high, high);
SET_WORD(bp_ptr, far_end_low_alarm_th_low, far_end_low_alarm_th_high, low);
```

With:

```
BP_WRITE_REG(bp_ptr, far_end_high_alarm_th_low, high);
BP_WRITE_REG(bp_ptr, far_end_high_alarm_th_high, 0x00);
BP_WRITE_REG(bp_ptr, far_end_low_alarm_th_low, low);
BP_WRITE_REG(bp_ptr, far_end_low_alarm_th_high, 0x00);
```

2.2.2 Increase DC Offset Meter Interval

On both the HTU-C and HTU-R, the software cancels out any DC offset on the DSL by calling the _DcCancel() function during the early stage of the start-up procedure. In previous versions of software the DC measurement is less accurate when a signal was present at the receiver. Hypothetically, a larger DC offset could make it more difficult to detect LOS (Loss Of Signal) on and off detections. Lab tests indicated that increasing the length of the DC measurement interval creates a more accurate measurement. Conexant has not seen any problem with LOS on or off detections, but the DC measurement interval was increased in order to decrease the variation in the DC error, and to further prevent any possible issues.

The meter interval for the DC measurement was increased from 8,192 symbols to 32,768 symbols. This increased interval causes the variation in the DC measurement to be decreased by about 4 times. For example, on one RS8973 device, the variation in the DC offset measurement was reduced from +/- 32 to +/- 8.

Modify SUC.C – Starting at Line 157

```
case ACTIVATE_SYSTEM: /* Activate startup process */
....
    _BtInitialize(no); /* Initialize bit-pump status for startup */
/*Addition starts*/
line 157 _SetMeterTimer(no, NORMAL_METER);
        /*Changed ALT_METER to NORMAL_METER to get a more accurate DC Offset*/
/*Addition ends*/
....

case DC_CANCELLATION:
    TIMER_BREAK(meter);

    _DcCancel(no);
/*Addition starts*/
/*
 * LOS (INIT_STARTUP)meter interval requires the ALT_METER
 * to be used.
 */
```

```
line 188  _SetMeterTimer(no, ALT_METER);  
/*Addition ends*/
```

Modify SUR.C – Starting at Line 146

```
    _BtInitialize(no); /* Initialize bit-pump status for startup */  
/*Addition starts*/  
line 146  _SetMeterTimer(no, NORMAL_METER);  
    /*Changed ALT_METER no NORMAL_METER to get a more accurate DC Offset*/  
/*Addition ends*/
```

....

```
case DC_CANCELLATION:  
    TIMER_BREAK(meter);  
  
    _DcCancel(no);  
/*Addition starts*/  
    /*  
     * LOS (WAIT_FOR_SIGNAL)meter interval requires the ALT_METER  
     * to be used.  
     */  
line 168  _SetMeterTimer(no, ALT_METER);  
/*Addition ends*/  
  
    _SetFelmMask(no);
```

3 Application and Channel Unit Code

3.1 Bug Fixes

3.1.1 _CU_SET_MFRAME API Does Not Process Zero Correctly

The _CU_SET_MFRAME API command sets the multi-frame length minus 1. The code does not properly handle a parameter value of zero that sets the MFRAME_SYNC pulse to be 125 μ s. Change CU_API.C - Starting at Line 570:

```
case _CU_SET_MFRAME:
/*
 * parameter is valid from 0x00 to 0x2F ( 0 to 47 ).
 * parameter = MF_LEN;
 * TMF_LOC = MF_LEN - 1;
 * ( MF_CNT + 1 ) * ( MF_LEN + 1 ) = 48;
 */
if ( parameter < 0 || parameter > _6MS_MFRAME )
{
break;
}
/*Addition starts*/
if ((parameter == 0x00)
{
common_wr_ptr->tmf_loc = 0x00;
}
else
{
/*Addition ends*/
common_wr_ptr->tmf_loc = parameter - 1;
/*Addition starts*/
}
/*Addition ends*/

common_wr_ptr->mf_len = parameter;
common_wr_ptr->mf_cnt = (48 / ( parameter + 1 )) - 1;
break;
```

3.1.2 LEDs not modified correctly when _NO_OF_LOOPS is less than 3.

The definition of the size of the arrays, cu_bp_led_block[] and cu_led_block[], is based on _NO_OF_LOOPS, but they are initialized based on hard-coded array indices. When _NO_OF_LOOPS is less than 3, the initialization of these arrays corrupts other RAM data.

Change CU_LED.C - Starting at Line 35:

CU_BP_LED_BLOCK BP_XDATA cu_bp_led_block[_NO_OF_LOOPS];

To:

CU_BP_LED_BLOCK BP_XDATA cu_bp_led_block[2];

Change CU.H - Starting at Line 1259:

extern CU_BP_LED_BLOCK BP_XDATA cu_bp_led_block[_NO_OF_LOOPS];

To:

extern CU_BP_LED_BLOCK BP_XDATA cu_bp_led_block[2];

Change CU_INIT.C - Starting at Line 466:

cu_led_block[0].reg = 0;

```
cu_led_block[1].reg = 0;
cu_led_block[2].reg = 0;
```

To:

```
for ( loop_cntr = 0 ; loop_cntr < _NO_OF_LOOPS ; loop_cntr++ )
{
    cu_led_block[loop_cntr].reg = 0;
}
```

3.1.3 T1/E1 Framer Unframed Mode API fix.

The _FRAMER_FRAME_FORMAT API command did not use the parameter values for Bt8360 or Bt8510 parts. With no parameter input, the command does nothing.

Change FRMR_API.C – Starting at Line 181

```
fr8510_mode_ptr->control_reg1.inhibit_abort_reframing = _CuFlags._CuFrameFormat ? 1 : 0;
```

To

```
fr8510_mode_ptr->control_reg1.inhibit_abort_reframing = parameter;
```

Change FRMR_API.C – Starting at Line 190

```
fr8360_mode_ptr->configuration.enable_esf_mode = _CuFlags._CuFrameFormat ? 0 : 1;
```

To

```
fr8360_mode_ptr->configuration.enable_esf_mode = !parameter;
```

3.2 Enhancements

Added support for the RS8370 to handle either framed or unframed data. Previously only framed data was handled.

Added FRMR_API.C – Starting at Line 199

```
case _BT8370:
#ifdef BT8370_FRAMER
    /*
     * Disable Receive Framer
     */
    fr8370_mode_ptr->recr_conf.rabort = parameter;
#endif /* BT8370_FRAMER */
    break;
```