# 【1】 UINT32 SetFreq(UINT32 FLO, UINT8 refClkType, UINT8 lpfBW)

{

UINT32 lpcnt=0;

UINT8 lock,adcout,vcorg;

UINT8 \_temper;

int PLLR=1;

/\*added by Dennis at PM 3:47 2008-08-20 v1.7.7\*/

UINT32 pll\_mul, r\_div;

UINT32 tmp;

UINT32 plln, pllf;

STTBX\_Print((L"ADMTV102 Init........\n"));

## 【1-1】ADMTV102\_Init(FLO, refClkType); /\*EVB\_DEFAULT\_REF\_CLOCK\*/

STTBX\_Print((L"ADMTV102 Set LPF........\n"));

## 【1-2】SetLPF(lpfBW, refClkType); /\*5=8MHz BW\*/ /\*added at PM5:22 2008-01-24 v1.7.2\*/

switch(refClkType)

{

case REFCLK13000: ADMTV102\_REFCLK = 13000;

break;

case REFCLK16384: ADMTV102\_REFCLK = 16384;

break;

case REFCLK19200: ADMTV102\_REFCLK = 19200;

break;

case REFCLK20480: ADMTV102\_REFCLK = 20480;

break;

case REFCLK24576:

{

ADMTV102\_REFCLK = 24576;

PLLR=2;

## 【1-3】WriteIIC\_RegData(25, PLLR, 0, 4); /\*Ref. Clock Divider\*/

}

break;

case REFCLK26000:

{

ADMTV102\_REFCLK = 26000;

#ifdef TEST\_MODE\_R26MHz

PLLR=2;

WriteIIC\_RegData(25, PLLR, 0, 4); 【1-3】 /\*Ref. Clock Divider\*/

#endif

}

break;

case REFCLK30400:

{

ADMTV102\_REFCLK = 30400; /\*revise point for support to 30.4MHz Ref. clock\*/

/\*SetLPF(REFCLK30400);\*/ /\*removed at PM5:22 2008-01-24 v1.7.2\*/

PLLR=2;

WriteIIC\_RegData(25, PLLR, 0, 4); 【1-3】 /\*Ref. Clock Divider\*/

}

break;

/\*case REFCLK36000: ADMTV102\_REFCLK = 36000;

break;\*/

case REFCLK38400:

{

ADMTV102\_REFCLK = 38400;

PLLR=2;

WriteIIC\_RegData(25, PLLR, 0, 4); 【1-3】/\*Ref. Clock Divider\*/

}

break;

case REFCLK20000: ADMTV102\_REFCLK = 20000; /\*Extra Ref. Clock Option.\*/

break;

}

STTBX\_Print((L"ADMTV102 Set PLL........\n"));

## 【1-4】PLLFREQ = LO2PLL\_Freq ( FLO );

g\_curFreq=FLO;

PLLN = ( PLLFREQ / ADMTV102\_REFCLK \* PLLR);

tmp=PLLFREQ-PLLN\*ADMTV102\_REFCLK / PLLR;

if (ADMTV102\_REFCLK==13000)

{

pll\_mul=2;

r\_div=8;

}

else

{

pll\_mul=1;

r\_div=16;

}

PLLF = ( (tmp \* 1024 \* 64) / (ADMTV102\_REFCLK/r\_div) \* PLLR \* pll\_mul);

plln=(int)( PLLFREQ / ADMTV102\_REFCLK \* PLLR);

pllf=(int)((( (float)PLLFREQ / ADMTV102\_REFCLK \* PLLR)-plln)\*1024\*1024);

/\*STTBX\_Print(("dif =%06x\t",abs((long)(PLLF-pllf))));

STTBX\_Print(("fixedp =%06x\t",PLLF));

STTBX\_Print(("real =%06x\n",pllf));\*/

## 【1-1-1-0-0】DATA47 = ReadIIC(47) & 0x1f;

if (PLLN<=66)

{/\*4-prescaler\*/

PC4 = 1;

PC8\_16= 0;

DATA47= DATA47 | (PC4<<6) | (PC8\_16<<5);

}

else

{/\*8-prescaler\*/

PC4 = 0;

PC8\_16= 0;

DATA47= DATA47 | (PC4<<6) | (PC8\_16<<5);

}

## 【1-1-1-01】WriteIIC( 39/\*0x27\*/, 0x00 ); /\*added : v1.6.3 at PM 20:39 2007-07-26\*/

/\*PLL Reset Enable\*/

WriteIIC( 47/\*0x2f\*/, DATA47 ); 【1-1-1-01】 /\*Reset Seq. 0 => 1 : //updated : v1.6.3 at PM 20:39 2007-07-26\*/

WriteIIC( 47/\*0x2f\*/, DATA47 | 0x80 ); 【1-1-1-01】

if(( PLLFREQ \* 2 ) < 2592000 )

VCOSEL = 0;

else

VCOSEL = 1;

/\*( ( PLLFREQ \* 2 ) < 2592000 ) ? VCOSEL=0 : VCOSEL = 1; \*/

/\*VCO Freq 2630000 =>2592000 : //changed previous code style : v1.7.2 at PM 20:39 2008-01-24\*/

## 【1-5】\_temper=ReadIIC\_RegData( 0x09, 0, 5) ; /\*To get Temperature sensor value \*/

STTBX\_Print((L"ADMTV102 dpPhaseTuning........\n"));

## 【1-6】dpPhaseTuning(FLO, \_temper); /\*To increase PhaseNoise Feature.\*/

/\*PLL Setting\*/

## 【1-1-1-01】WriteIIC( 27/\*0x1b\*/, (UINT8)( PLLN & 0xFF ) );

if (FLO>400000) /\*f iRF=UHF updated : v1.7.2 at 2008-01-29\*/

WriteIIC(26, (g\_icp<<2)|(UINT8)(( PLLN&0x300 ) >> 8 )); 【1-1-1-01】

else /\*if iRF=VHF\*/

WriteIIC(26, (UINT8)(0xFC | ( ( PLLN&0x300 ) >> 8 ))); 【1-1-1-01】/\*updated : v1.7.2 at 2008-01-29\*/

/\* WriteIIC( 26, (g\_icp<<2) | ( ( PLLN&0x300 ) >> 8 ) );\*/

WriteIIC( 30/\*0x1e\*/, (UINT8)(PLLF & 0xFF) ); 【1-1-1-01】

WriteIIC( 29/\*0x1d\*/, (UINT8)(( PLLF & 0x0FF00 ) >> 8));

WriteIIC( 28/\*0x1c\*/, (UINT8)(( DIVSEL << 4 ) | ( VCOSEL << 7 ) | ( ( PLLF & 0xF0000 ) >> 16 ))); 【1-1-1-01】

/\*Reset Seq. 1 => 0 updated : v1.6.3 at PM 20:39 2007-07-26\*/

WriteIIC( 47/\*0x2f\*/, DATA47 ); 【1-1-1-01】

/\*updated : v1.6.4 at 2007-08-10\*/

WriteIIC( 39/\*0x27\*/, (UINT8)g\_convco ); 【1-1-1-01】

if (FLO>400000) /\*if iRF=UHF //updated : v1.7.2 at 2008-01-29\*/

{

WriteIIC( 41/\*0x29\*/, 0xBF); 【1-1-1-01】

}

Sleep(1);/\*Sleep(1)\*/

fChn\_ICP=0; /\*added in order to initialize a switch flag for ICP register :\*/

/\*updated : v1.7.5 at 2008-04-22\*/

/\*you should change this, Mandatory, to prevent performance decreasing\*/

while(1)

{

## 【1-1-1-0-0】lock = ( ReadIIC ( 6 ) >> 1 )&0x01;

adcout = ReadIIC ( 4 ) 【1-1-1-0-0】& 0x0f;

if ( ( ( adcout > 2 ) && ( adcout < 12 ) ) && lock ) /\*pll lock cross check\*/

{

break; /\*pll lock\*/

}

Sleep(5); /\*5us delay\*/

/\*

v1.7.5 Mandatory Update Point :START

you should change this, Mandatory, to prevent performance decreasing

\*/

if (lpcnt++>=20)/\* return -1; \*/ /\* pll unlock\*/

{/\*return -1;\*/

【1-1-1-0-0】vcorg=(ReadIIC(0x0d)&0x1F);

if ((adcout<=2)&&(vcorg==0)&&(VCOSEL==1))

{

## 【1-3】WriteIIC\_RegData(28, 0, 7, 1); /\*VCOSEL=0\*/

g\_icp=0x1F;

WriteIIC\_RegData(26, g\_icp, 2,6); 【1-3】

fChn\_ICP=1;

}

else

if ((adcout>=12)&&(vcorg==0x1F)&&(VCOSEL==0))

{

WriteIIC\_RegData(28, 1, 7, 1); 【1-3】/\*VCOSEL=1\*/

g\_icp=0x3F;

WriteIIC\_RegData(26, g\_icp, 2,6); 【1-3】

fChn\_ICP=1;

}

lpcnt=0;

while(1)

{

## 【1-5】lock = ReadIIC\_RegData(6,1,1);

adcout = ReadIIC\_RegData(4,0,4); 【1-5】

if (((adcout>2)&&(adcout<12)) && lock)

{

break;

}

Sleep(5); /\*5us delay\*/

if (lpcnt++>=20) return -1;

}

}

/\*

v1.7.5 Mandatory Update Point : END

\*/

}

return FLO;

}

# 【1-1】static void ADMTV102\_Init(UINT32 FLO, UINT8 refClkType)

{

if (g\_curBand==0xff) /\*changed according to modified init value at v1.7.3\*/

if (FLO>400000)

**【1-1-1】ADMTV102\_Init\_UHF(refClkType);**

else

**【1-1-2】ADMTV102\_Init\_VHF(refClkType);**

else

{

if (g\_curBand==VHF\_BAND)

if (FLO>400000)

**【1-1-1】ADMTV102\_Init\_UHF(refClkType);**

if (g\_curBand==UHF\_BAND)

if (FLO<400000)

**【1-1-2】ADMTV102\_Init\_VHF(refClkType);**

}

}

# 【1-2】static void SetLPF(UINT8 nBWSel, UINT8 refClkType) /\*updated : v1.7.2\*/

{

UINT8 tuneval;

**【1-1-1-01】WriteIIC( 0x15 , 0x38 | (nBWSel&0x07)); /\* LPF BW selection, revised at PM5:22 2008-01-24** v1.7.2\*/

WriteIIC( 0x25 , (CTUNEOFS<<3) | (\_EXTUNEON << 2) | (\_TUNEEN<<1) ); 【1-1-1-01】

WriteIIC( 0x25 , (CTUNEOFS<<3) | (\_EXTUNEOFF << 2) | (\_TUNEEN<<1) ); 【1-1-1-01】

Sleep(10);

## 【1-1-1-0-0】tuneval=ReadIIC(15); /\*/CTUNE val\*/

**【1-1-1-01】WriteIIC( 0x25 , (CTUNEOFS<<3) | (\_EXTUNEON << 2) | (\_TUNEEN << 1) ); /\*change Tuning** mode : auto-tune => manual tune(hold mode).\*/

if (refClkType==REFCLK30400)

WriteIIC( 0x25 , ((tuneval+CTUNE\_CLKOFS)<<3) | (\_EXTUNEON << 2) | (\_TUNEEN << 1) ); 【1-1-1-01】 /\*Write CTUNE val. in order to store tuned value.\*/

else

WriteIIC( 0x25 , (tuneval<<3) | (\_EXTUNEON << 2) | (\_TUNEEN << 1) ); 【1-1-1-01】 /\*Write CTUNE val. in order to store tuned value.\*/

}

# 【1-4】static UINT32 LO2PLL\_Freq(UINT32 lofreq)

{

UINT32 pllfreq=0;

UINT32 fdefHiBoundFreq=1880000; /\*Hi Boundary Freq 2000000 => 1880000 : \*/

/\*updated : v1.5 at PM 20:39 2007-06-8\*/

UINT32 fdefLoBoundFreq=940000;/\*Lo Boundary Freq 1000000 => 940000 :\*/

/\*updated : v1.5 at PM 20:39 2007-06-8\*/

{

if ((lofreq < fdefHiBoundFreq)&&(lofreq>=fdefLoBoundFreq))

DIVSEL=4;

else

if ((lofreq < fdefHiBoundFreq/2)&&(lofreq>=fdefLoBoundFreq/2))

DIVSEL=3;

else

if ((lofreq < fdefHiBoundFreq/4)&&(lofreq>=fdefLoBoundFreq/4))

DIVSEL=2;

else

if ((lofreq < fdefHiBoundFreq/8)&&(lofreq>=fdefLoBoundFreq/8))

DIVSEL=1;

else

if ((lofreq < fdefHiBoundFreq/16)&&(lofreq>=fdefLoBoundFreq/16))

DIVSEL=0;

switch(DIVSEL)

{

case 0:

pllfreq = lofreq\*16;

return pllfreq;

case 1:

pllfreq = lofreq\*8;

return pllfreq;

case 2:

pllfreq = lofreq\*4;

return pllfreq;

case 3:

pllfreq = lofreq\*2;

return pllfreq;

case 4:

pllfreq = lofreq\*1;

return pllfreq;

default:

pllfreq = lofreq\*2;

return pllfreq;

}

}

return pllfreq;

}

# 【1-6】static void dpPhaseTuning(INT32 lofreq, INT8 temper)

{

if (fDegVcoApply)

{

if (g\_curBand==UHF\_BAND)

if (temper<=vlowDegBoundary) /\*low boundary\*/

{

g\_convco=rglowDegCONVCO;

g\_icp=0x3F;

g\_curTempState=LOW\_TEMP;

}

else

if (temper>=vhighDegBoundary) /\*high boundary\*/

{

g\_convco=rgHighDegCONVCO;

if (fChn\_ICP==0) /\*(ICP value would be changed according to fChn\_ICP value : updated : v1.7.5 at 2008-04-22

you should change this, Mandatory, to prevent performance decreasing\*/

if (( lofreq > 610000 ) && ( lofreq < 648000 )) /\*610MHz ~ 648MHz\*/

g\_icp=0x3F;//FROM 1F TO 3F CARY

else

g\_icp=0x3F;

else

if (( lofreq > 610000 ) && ( lofreq < 648000 )) /\*610MHz ~ 648MHz\*/

g\_icp=0x1F;//FROM 3F TO 1F CARY

else

g\_icp=0x1F;

g\_curTempState=HIGH\_TEMP;

}

else

{

if (g\_curTempState)

{

g\_convco=rglowDegCONVCO;

g\_icp=0x3F;

}

else

{

g\_convco=rgHighDegCONVCO;

if (fChn\_ICP==0) /\*ICP value would be changed according to fChn\_ICP value : updated : v1.7.5 at 2008-04-22

you should change this, Mandatory, to prevent performance decreasing\*/

if (( lofreq > 610000 ) && ( lofreq < 648000 )) /\*610MHz ~ 648MHz\*/

g\_icp=0x3F;//FROM 1F TO 3F CARY

else

g\_icp=0x3F;

else

if (( lofreq > 610000 ) && ( lofreq < 648000 )) /\*610MHz ~ 648MHz\*/

g\_icp=0x1F;//FROM 3F TO 1F CARY

else

g\_icp=0x1F;

}

}

else

{

if (temper<=vlowDegBoundary) /\*low boundary\*/

{

g\_convco=rglowDegCONVCO\_VHF;

g\_curTempState=LOW\_TEMP;

}

else

if (temper>=vhighDegBoundary) /\*high boundary\*/

{

g\_convco=rgHighDegCONVCO;

g\_curTempState=HIGH\_TEMP;

}

}

}

}

# 【2】UINT8 ADMTV\_Test(void)

{

UINT32 CTUNEVAL;

UINT16 CHIPID;

UINT8 ADCOUT;

UINT8 lock;

unsigned int a,b,c;

unsigned int Reg1,Reg2,Reg3,Reg4,Reg5,Reg6,Reg7,Reg8;

UINT8 \_temper;

while(1)

{

Sleep(1000);

## 【1-1-1-0-0】CHIPID = (UINT16)(ReadIIC(1)<<8)|(ReadIIC(2));

lock=(ReadIIC(0x06)>>1)&0x01; 【1-1-1-0-0】

ADCOUT=(ReadIIC(0x04)&0x0f); 【1-1-1-0-0】

CTUNEVAL=(ReadIIC(0x0f)); 【1-1-1-0-0】

a=ReadIIC(0x1E); 【1-1-1-0-0】

b=ReadIIC(0x1D)\*256; 【1-1-1-0-0】

c=((ReadIIC(0x1C)&0x0f)\*256\*256); 【1-1-1-0-0】

Reg1 = ReadIIC(0x07); 【1-1-1-0-0】

Reg2 = ReadIIC(0x0D); 【1-1-1-0-0】

Reg3 = ReadIIC(0x05); 【1-1-1-0-0】

Reg4 = ReadIIC(0x06); 【1-1-1-0-0】

Reg5 = ReadIIC(0x17); 【1-1-1-0-0】

Reg6 = ReadIIC(0x47); 【1-1-1-0-0】

Reg7 = ReadIIC(0x48); 【1-1-1-0-0】

Reg8 = ReadIIC(0x54); 【1-1-1-0-0】

STTBX\_Print((L"07->0x%02x 0D-> 0x%02x 05-> 0x%02x 06-> 0x%02x \n",Reg1,Reg2,Reg3,Reg4));

STTBX\_Print((L"17->0x%02x 47-> 0x%02x 48-> 0x%02x 54-> 0x%02x \n",Reg5,Reg6,Reg7,Reg8));

STTBX\_Print((L"CHIP ID = ADMTV%3x ", CHIPID));

STTBX\_Print((L"lock = %3x ", lock));

STTBX\_Print((L"ADCOUT = %2x, ", ADCOUT));

STTBX\_Print((L"CTUNE Value = %2x, ", CTUNEVAL));

STTBX\_Print((L"PLLF = %d\n ", a+b+c));

\_temper=ReadIIC\_RegData( 0x09, 0, 5); 【1-5】/\*To get Temperature sensor value \*/

STTBX\_Print((L"Temperature 0x%02x\n", \_temper));

break;

}

return lock;

}

# 【1-1】static void ADMTV102\_Init(UINT32 FLO, UINT8 refClkType)

{

if (g\_curBand==0xff) /\*changed according to modified init value at v1.7.3\*/

if (FLO>400000)

## 【1-1-1】ADMTV102\_Init\_UHF(refClkType);

else

## 【1-1-2】ADMTV102\_Init\_VHF(refClkType);

else

{

if (g\_curBand==VHF\_BAND)

if (FLO>400000)

## 【1-1-1】ADMTV102\_Init\_UHF(refClkType);

if (g\_curBand==UHF\_BAND)

if (FLO<400000)

## 【1-1-2】ADMTV102\_Init\_VHF(refClkType);

}

}

# 【1-1-1】static void ADMTV102\_Init\_UHF(UINT8 refClkType) /\*updated : v1.6.8 at PM 16:48 2007-11-23\*/

{

/\*~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Mandatory Update Start Point ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\*/

g\_curBand=UHF\_BAND; /\* added at v1.7.2 at PM 16:16 2008-01-29\*/

## 【1-1-1-0】GetUpdateTAG();/\* Get Chip Revision Info. and then redifine macro values.\*/

/\*ADMTV102\_rev0.4 ---> 21 Jan,2008\*/

/\*-----------------ADDR: 10 , DEF: 49 , MOD: 4c / Changed LNA current \*/

## 【1-1-1-01】WriteIIC( 0x10, 0x6A);

/\*-----------------ADDR: 11 , DEF: d2 , MOD: d8 / Changed Mixer current \*/

WriteIIC( 0x11, 0xD8); 【1-1-1-01】

/\*-----------------ADDR: 12 , DEF: 40 , MOD: c0 / Changed Mixer gain \*/

WriteIIC( 0x12, 0xC0); 【1-1-1-01】

/\*-----------------ADDR: 15 , DEF: 25 , MOD: 3d / changed power detector gain \*/

/\*WriteIIC( 0x15, 0x3D);\*/

/\*-----------------ADDR: 17 , DEF: c8 , MOD: 97 / Changed TOP and ADJ to optimize SNR and ACI \*/

WriteIIC( 0x17, 0x87) 【1-1-1-01】; //revised at ADMTV102\_UHF\_Rev0\_4.txt

/\*-----------------ADDR: 18 , DEF: b9 , MOD: 03 / Changed power detector saturation voltage point and warning voltage point \*/

WriteIIC( 0x18, 0x03); 【1-1-1-01】 //Inserted at 2007.08.10 ADMTV102\_rev0.3.txt

/\*-----------------ADDR: 1A , DEF: 28 , MOD: Fc / Changed charge pump current to optimize phase noise \*/

/\*WriteIIC( 0x1A, 0xFC); Modifed at 2007.08.10 ADMTV102\_rev0.3.txt\*/

/\*-----------------ADDR: 1B , DEF: 20 , MOD: 51 / Changed PLL N because of VCO frequency change \*/

/\*WriteIIC( 0x1B, 0x51); \*/

/\*----------------ADDR: 1C , DEF: 30 , MOD: b4 / Changed VCOSEL,PLLF \*/

/\*WriteIIC( 0x1C, 0xB4);\*/

/\*----------------ADDR: 1D , DEF: 00 , MOD: bd / Changed PLL F because of VCO frequency change \*/

/\*WriteIIC( 0x1D, 0xA4);\*/

/\*----------------ADDR: 1E , DEF: 80 , MOD: f9 / PLL setting value \*/

/\*WriteIIC( 0x1E, 0xFB);\*/

/\*----------------ADDR: 1F , DEF: 16 , MOD: 17 / Changed RFPGA amp current \*/

WriteIIC( 0x1F, 0x17); 【1-1-1-01】

/\*----------------ADDR: 20 , DEF: d2 , MOD: ff / Changed RFPGA amp current \*/

WriteIIC( 0x20, 0xFF); 【1-1-1-01】

/\*----------------ADDR: 21 , DEF: 6e , MOD: a4 / Changed RFPGA amp current \*/

WriteIIC( 0x21, 0xA4); 【1-1-1-01】

/\*----------------ADDR: 22 , DEF: 6e , MOD: a4 / Changed RFPGA amp current \*/

WriteIIC( 0x22, 0xA4); 【1-1-1-01】

/\*----------------ADDR: 23 , DEF: dc , MOD: df / Changed Mixer Bias contol \*/

WriteIIC( 0x23, 0xDF); 【1-1-1-01】

/\*----------------ADDR: 25 , DEF: 02 , MOD: 06 / Changed LPF cutoff frequency and Load control\*/

/\*WriteIIC( 0x25, 0x06);\*/

/\*----------------ADDR: 26 , DEF: ee , MOD: fa / Changed PLL BUFFER CURRENT \*/

WriteIIC( 0x26, 0xFA); 【1-1-1-01】

/\*----------------ADDR: 27 , DEF: 99 , MOD: 00 / Changed CONVCOL/H for VCO current control \*/

WriteIIC( 0x27, 0x00); 【1-1-1-01】

/\*----------------ADDR: 28 , DEF: 99 , MOD: ff / Changed CONVCOBUFL/H for VCO buffer amplifier current control \*/

WriteIIC( 0x28, 0xFF); 【1-1-1-01】

/\*----------------ADDR: 29 , DEF: ee , MOD: ff / Changed CONDIV1/2 for first and second divider current control \*/

WriteIIC( 0x29, 0xFF); 【1-1-1-01】

/\*----------------ADDR: 2A , DEF: ee , MOD: ff / Changed CONDIV3/4 for third and last divider current control \*/

WriteIIC( 0x2A, 0xFF); 【1-1-1-01】

/\*----------------ADDR: 2B , DEF: ef , MOD: e7 / Changed CONDIV5 and CONBUF for third and last divider current control\*/

WriteIIC( 0x2B, 0xE7); 【1-1-1-01】

/\*----------------ADDR: 2C , DEF: ec , MOD: fe / Changed CONBUF0/1 for first Buffer amp current control \*/

WriteIIC( 0x2C, 0xFE); 【1-1-1-01】

/\*----------------ADDR: 2E , DEF: fe , MOD: fb / Changed CONBUF4 for forth Buffer amp current control \*/

WriteIIC( 0x2E, 0xFB); 【1-1-1-01】

/\*----------------ADDR: 30 , DEF: 01 , MOD: 80 / Changed LFSW(Internal Loop Filter) to improve phase noise \*/

WriteIIC( 0x30, 0x80); 【1-1-1-01】 /\*revised at ADMTV102\_UHF\_Rev0\_4.txt \*/

/\*----------------ADDR: 31 , DEF: 44 , MOD: 00 / Changed VCO serching speed (Fast) \*/

WriteIIC( 0x31, 0x04); 【1-1-1-01】/\*0x00 => 0x04 \*//\*modified at PM5:22 2008-03-12 v1.7.4\*/

/\*----------------ADDR: 32 , DEF: b2 , MOD: CD / Changed LOOP filter boundary \*/

WriteIIC( 0x32, 0xC2); 【1-1-1-01】

/\*----------------ADDR: 33 , DEF: 90 , MOD: 80 / DC offset control \*/

WriteIIC( 0x33, 0x80); 【1-1-1-01】

/\*----------------ADDR: 34 , DEF: e1 , MOD: ec / DC offset control \*/

WriteIIC( 0x34, 0xEC); 【1-1-1-01】

/\*----------------ADDR: 43 , DEF: 7f , MOD: 3f \*//\* TSPD(Time Slicing Power Down) control \*/

WriteIIC( 0x43, 0x3F); 【1-1-1-01】/\*Modifed at 2007.08.10 ADMTV102\_rev0.3.txt\*/

/\*----------------ADDR: 44 , DEF: ff , MOD: f8 \*/ /\* TSPD(Time Slicing Power Down) control \*/

WriteIIC( 0x44, 0xF8); 【1-1-1-01】/\*Modifed at 2007.08.10 ADMTV102\_rev0.3.txt\*/

/\*----------------ADDR: 48 , DEF: 82 , MOD: 21 / Changed current for output buffer amp \*/

WriteIIC( 0x48, 0x21); 【1-1-1-01】//for SNR CARY

/\*----------------ADDR: 49 , DEF: 88 , MOD: 08 / Changed gain mode for output buffer amp \*/

WriteIIC( 0x49, 0x08); 【1-1-1-01】

/\*----------------ADDR: 4A , DEF: b4 , MOD: a0 / Changed trip point for BBVGA \*/

WriteIIC( 0x4A, 0xA0); 【1-1-1-01】

/\*----------------ADDR: 4B , DEF: 9b , MOD: 9d / Changed trip point for RFPGA \*/

WriteIIC( 0x4B, 0x9D); 【1-1-1-01】

/\*----------------ADDR: 4C , DEF: 9b , MOD: 9d / Changed ADJRSSI warning point \*/

WriteIIC( 0x4C, 0x9D); 【1-1-1-01】

/\*----------------ADDR: 4D , DEF: cb , MOD: c3 / Changed PLL current to stable PLL lock \*/

WriteIIC( 0x4D, 0xC3); 【1-1-1-01】

/\*----------------ADDR: 24 , DEF: ad , MOD: ?F / Changed MPGAI2C to improve SNR \*/

switch(refClkType)

{/\*revised at ADMTV102\_UHF\_Rev0\_4.txt\*/

/\* bit [4:7] <CLKSELI2C = 0 : 13MHz, 1 : 16.384MHz, 2 : 19.2MHz, 3 : 20.48MHz, 4 : 24.576MHz, 5 : 26MHz, 6 : 30.4MHz, 7 : 38.4MHz>\*/

case REFCLK13000:

WriteIIC( 0x24, 0x0F); 【1-1-1-01】

break;

case REFCLK16384:

WriteIIC( 0x24, 0x1F); 【1-1-1-01】

break;

case REFCLK19200:

WriteIIC( 0x24, 0x2F); 【1-1-1-01】

break;

case REFCLK20480:

WriteIIC( 0x24, 0x3F); 【1-1-1-01】

break;

case REFCLK24576:

WriteIIC( 0x24, 0x4F); 【1-1-1-01】

break;

case REFCLK26000:

WriteIIC( 0x24, 0x5F); 【1-1-1-01】

break;

case REFCLK30400:

WriteIIC( 0x24, REFCLK30400\_CLKSEL\_REG); 【1-1-1-01】 //revised at v1.6.9

break;

case REFCLK38400:

WriteIIC( 0x24, 0x7F); 【1-1-1-01】

break;

case REFCLK20000: //Extra Ref. Clock Option.

WriteIIC( 0x24, 0x3F); 【1-1-1-01】

break;

}

/\*~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Mandatory Update End Point ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\*/

}

# 【1-1-2】static void ADMTV102\_Init\_VHF(UINT8 refClkType) //updated : v1.6.8 at PM 16:48 2007-11-23

{

/\*~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Mandatory Update Start Point ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\*/

g\_curBand=VHF\_BAND; // added at v1.7.2 at PM 16:16 2008-01-29

## 【1-1-1-0】GetUpdateTAG();// Get Chip Revision Info. and then redifine macro values.

/\*ADMTV102\_rev0.4 ---> 21 Jan,2008\*/

/\*----------------ADDR: 10 , DEF: 49 , MOD: 08 \*/

## 【1-1-1-01】WriteIIC( 0x10, 0x08 );

/\*----------------ADDR: 11 , DEF: d2 , MOD: d8 \*/

WriteIIC( 0x11, 0xc2 ); 【1-1-1-01】

/\*----------------ADDR: 12 , DEF: 40 , MOD: c0 \*/

WriteIIC( 0x12, 0xC0 ); 【1-1-1-01】

/\*----------------ADDR: 15 , DEF: 25 , MOD: 3d \*/

/\*----------------ADDR: 17 , DEF: c8 , MOD: 98 \*/

WriteIIC( 0x17, 0x98 ); 【1-1-1-01】 //revised at ADMTV102\_VHF\_Rev0\_4.txt

/\*----------------ADDR: 1F , DEF: 16 , MOD: 17 \*/

WriteIIC( 0x1F, 0x17 ); 【1-1-1-01】

/\*----------------ADDR: 20 , DEF: d2 , MOD: ff \*/

WriteIIC( 0x20, 0x9b ); 【1-1-1-01】

/\*----------------ADDR: 21 , DEF: 6e , MOD: a4 \*/

WriteIIC( 0x21, 0xA4 ); 【1-1-1-01】

/\*----------------ADDR: 22 , DEF: 6e , MOD: a4 \*/

WriteIIC( 0x22, 0xA4 ); 【1-1-1-01】

/\*----------------ADDR: 23 , DEF: dc , MOD: 9f \*/

WriteIIC( 0x23, 0x9F ); 【1-1-1-01】

/\*----------------ADDR: 25 , DEF: 02 , MOD: 6e \*/

/\*WriteIIC( 0x25, 0x6E );\*/

/\*----------------ADDR: 26 , DEF: ee , MOD: f9 \*/

WriteIIC( 0x26, 0xF9 ); 【1-1-1-01】

/\*----------------ADDR: 27 , DEF: 99 , MOD: 11 \*/

WriteIIC( 0x27, 0x11 ); 【1-1-1-01】

/\*----------------ADDR: 28 , DEF: 99 , MOD: 92 \*/

WriteIIC( 0x28, 0x92 ); 【1-1-1-01】

/\*----------------ADDR: 29 , DEF: ee , MOD: be \*/

WriteIIC( 0x29, 0xBC ); 【1-1-1-01】

/\*----------------ADDR: 2B , DEF: ef , MOD: e7 \*/

WriteIIC( 0x2B, 0xE7 ); 【1-1-1-01】

/\*----------------ADDR: 2D , DEF: ff , MOD: 9c \*/

WriteIIC( 0x2D, 0x9C ); 【1-1-1-01】

/\*----------------ADDR: 2E , DEF: fe , MOD: ce \*/

WriteIIC( 0x2E, 0xCE ); 【1-1-1-01】

/\*----------------ADDR: 2F , DEF: 10 , MOD: 1f \*/

WriteIIC( 0x2F, 0x1F ); 【1-1-1-01】

/\*----------------ADDR: 30 , DEF: 01 , MOD: 80 \*/

WriteIIC( 0x30, 0x80 ); 【1-1-1-01】 /\*revised at ADMTV102\_VHF\_Rev0\_4.txt\*/

/\*----------------ADDR: 31 , DEF: 44 , MOD: 00 \*/

WriteIIC( 0x31, 0x04 ); 【1-1-1-01】 /\*0x00 => 0x04 \*//\*modified at PM5:22 2008-03-12 v1.7.4\*/

/\*----------------ADDR: 32 , DEF: b2 , MOD: c2 \*/

WriteIIC( 0x32, 0xC2 ); 【1-1-1-01】

/\*----------------ADDR: 33 , DEF: 90 , MOD: 80 \*/

WriteIIC( 0x33, 0x80 ); 【1-1-1-01】

/\*----------------ADDR: 34 , DEF: e1 , MOD: ec \*/

WriteIIC( 0x34, 0xEC ); 【1-1-1-01】

/\*----------------ADDR: 47 , DEF: c2 , MOD: e2 \*/

/\*WriteIIC( 0x47, 0xE2 );\*/ /\*removed at v1.7.2 PM 2:50 2008-01-25 \*/

/\*----------------ADDR: 48 , DEF: 82 , MOD: 29 \*/

WriteIIC( 0x48, 0x29 ); 【1-1-1-01】

/\*----------------ADDR: 49 , DEF: 88 , MOD: 08 \*/

WriteIIC( 0x49, 0x08 ); 【1-1-1-01】

/\*----------------ADDR: 4A , DEF: b4 , MOD: a0 \*/

WriteIIC( 0x4A, 0xA0 ); 【1-1-1-01】

/\*----------------ADDR: 4B , DEF: 9b , MOD: 9d \*/

WriteIIC( 0x4B, 0x9D ); 【1-1-1-01】

/\*----------------ADDR: 4C , DEF: 9b , MOD: 9d \*/

WriteIIC( 0x4C, 0x9D ); 【1-1-1-01】

/\*----------------ADDR: 4D , DEF: cb , MOD: c3 \*/

WriteIIC( 0x4D, 0xC3 ); 【1-1-1-01】

/\*----------------ADDR: 24 , DEF: ad , MOD: ?F / Changed MPGAI2C to improve SNR \*/

switch(refClkType)

{/\*revised at ADMTV102\_VHF\_Rev0\_4.txt\*/

/\* bit [4:7] <CLKSELI2C = 0 : 13MHz, 1 : 16.384MHz, 2 : 19.2MHz, 3 : 20.48MHz, 4 : 24.576MHz, 5 : 26MHz, 6 : 30.4MHz, 7 : 38.4MHz>\*/

case REFCLK13000:

WriteIIC( 0x24, 0x0F); 【1-1-1-01】

break;

case REFCLK16384:

WriteIIC( 0x24, 0x1F); 【1-1-1-01】

break;

case REFCLK19200:

WriteIIC( 0x24, 0x2F); 【1-1-1-01】

break;

case REFCLK20480:

WriteIIC( 0x24, 0x3F); 【1-1-1-01】

break;

case REFCLK24576:

WriteIIC( 0x24, 0x4F); 【1-1-1-01】

break;

case REFCLK26000:

WriteIIC( 0x24, 0x5F); 【1-1-1-01】

break;

case REFCLK30400:

WriteIIC( 0x24, REFCLK30400\_CLKSEL\_REG); 【1-1-1-01】 /\*revised at v1.6.9 \*/

break;

/\*case REFCLK36000:

WriteIIC( 0x24, 0x6F);

break;

\*/

case REFCLK38400:

WriteIIC( 0x24, 0x7F); 【1-1-1-01】

break;

case REFCLK20000: //Extra Ref. Clock Option.

WriteIIC( 0x24, 0x3F); 【1-1-1-01】

break;

}

/\*~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Mandatory Update End Point ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\*/

}

# 【6】void doUserTimer(UINT8 nIDEvent) /\* event time : 1 ~ 5 seconds after frequency setting\*/

{

UINT8 \_temper;

switch(nIDEvent)

{

case N\_1000MS\_EVENT :

/\*KillTimer(m\_hWnd, N\_1000MS\_EVENT);\*/

## 【1-5】\_temper=ReadIIC\_RegData( 0x09, 0, 5) ; /\*To get Temperature sensor value \*/

## 【1-6】dpPhaseTuning(g\_curFreq, \_temper);

if (g\_curBand==UHF\_BAND)

## 【1-3】WriteIIC\_RegData(26/\*0x1A\*/, g\_icp, 2,6);

## 【1-1-1-01】WriteIIC( 39/\*0x27\*/, g\_convco );

/\*TRACE ("TIMER :FREQ = %f TEMP = %02d ICP = %02x DIVCON = %02x\n",m\_curFreq/2, \_temper, g\_icp, g\_convco);\*/

/\*SetTimer(N\_1000MS\_EVENT,1000,NULL);\*/

break;

}

}

# 【1-1-1-0】static UINT8 GetUpdateTAG()

{

UINT8 split\_id;

## 【1-1-1-0-0】split\_id=ReadIIC(0);

switch(split\_id)

{

case 0x0E: /\*support 36MHz Ref. Clock\*/

#undef CTUNE\_CLKOFS

#define CTUNE\_CLKOFS 9

#undef REFCLK30400\_CLKSEL\_REG

#define REFCLK30400\_CLKSEL\_REG 0x5F

break;

case 0x0F: /\*replace 36MHz Ref. Clock with 30.4MHz Ref. Clock\*/

#undef CTUNE\_CLKOFS

#define CTUNE\_CLKOFS 0

#undef REFCLK30400\_CLKSEL\_REG

#define REFCLK30400\_CLKSEL\_REG 0x6F

break;

}

return split\_id;

}

# 【8】static float GetRFRSSI(float fBBAGC\_INPUT\_VOLTAGE)

{

UINT8 rfagc;

UINT8 lnagain, gvbb;

## 【1-1-1-0-0】rfagc = ReadIIC(5) + (ReadIIC(6) & 0x01) << 8;

## 【1-1-1-0-0】lnagain= (ReadIIC(0x0d) & 0x60) >> 5;

## 【1-1-1-0-0】gvbb=(ReadIIC(4) & 0xf0) >> 4;

return -(rfagc\*0.125)-(lnagain\*8.3)-3\*(gvbb-5)-10\*(fBBAGC\_INPUT\_VOLTAGE-0.5);

}

# 【1-3】static void WriteIIC\_RegData(UINT8 Taddr, UINT8 Tdata, UINT8 bitposition, UINT8 bitcount)

{

UINT8 Data;

UINT8 bitmask,i,bitmask2;

bitmask = 0xff;

bitmask2 = 0x0;

for(i=bitposition;i<bitposition+bitcount;i++) {

bitmask = bitmask -(int)pow(2.,i);

bitmask2 = bitmask2+(int)pow(2.,i);

}

bitmask2 = bitmask2/pow(2.0,bitposition);

Tdata = Tdata&bitmask2;

## 【1-1-1-0-0】Data = ReadIIC(Taddr);

Data = (Data&bitmask)+(Tdata<<bitposition);

## 【1-1-1-01】WriteIIC(Taddr, Data);

}

# 【1-1-1-01】static void WriteIIC(UINT8 bIndex, UINT8 bData)

{

UINT8 WriteData[2];

WriteData[0] = bIndex;

WriteData[1] = bData;

## 【API】Write\_I2C(ADMTV102\_I2C\_ADDRESS,2,WriteData);

}

# 【1-5】static UINT8 ReadIIC\_RegData(UINT8 Taddr, UINT8 bitposition, UINT8 bitcount)

{

UINT8 Data;

UINT8 bitmask,i;

bitmask=0x00;

for(i=bitposition;i<bitposition+bitcount;i++) {

bitmask=bitmask+(int)pow(2.,i);

}

## 【1-1-1-0-0】Data=ReadIIC(Taddr);

Data=(Data&bitmask)>>bitposition;

return Data;

}

# 【1-1-1-0-0】static UINT8 ReadIIC(UINT8 bIndex)

{

UINT8 ReadData;

## 【API】Write\_I2C(ADMTV102\_I2C\_ADDRESS,1,&bIndex);

## 【API】Read\_I2C(ADMTV102\_I2C\_ADDRESS,1,&ReadData);

return ReadData;

}