

HC32L110 series

32 -bit ARM ® Cortex ® -M0+ microcontroller

User manual

statement

- Huada Semiconductor Co., Ltd. (hereinafter referred to as "HDSC") reserves the right to change, correct, enhance and modify Huada Semiconductor products and/or the rights of this document without notice. Users can obtain the latest relevant information before placing an order. HDSC products are in accordance with the basic purchase and sale contract. Sales are carried out with stated terms and conditions of sale.
- Customers should select suitable HDC products for your application, and design, verify and test your application to ensure that your application meets the relevant requirements. Comply with standards and any safety, security or other requirements. The customer shall be solely responsible for this.
- HDSC hereby confirms that it has not granted any intellectual property licenses expressly or implicitly.
- For the resale of HDSC products, if the terms are different from those stipulated here, any warranty commitments made by HDSC for such products are invalid.
- Any graphics or words with the “®” or “™” logo are trademarks of HDC. All other products or services shown on the HDC product names are the property of their respective owners.
- The information in this notice replaces and replaces the information in the previous version.

©2020 Huada Semiconductor Co., Ltd.-All rights reserved

contents

<u>statement</u>	2
<u>contents</u>	3
<u>Introduction</u>	twenty three

<u>1 System structure.....</u>	<u>twenty four</u>
<u>1.1 Overview.....</u>	<u>twenty four</u>
<u>1.2 System address division.....</u>	<u>25</u>
<u>1.3 Memory and module address allocation.....</u>	<u>27</u>
<u>2 Working mode.....</u>	<u>28</u>
<u>2.1 Operating mode.....</u>	<u>30</u>
<u>2.2 Sleep mode.....</u>	<u>31</u>
<u>2.3 Deep sleep mode.....</u>	<u>33</u>
<u>3 System Controller (SYSCTRL)</u>	<u>36</u>
<u>3.1 Clock source introduction.....</u>	<u>36</u>
<u>3.1.1 Internal high-speed RC clock RCH.....</u>	<u>37</u>
<u>3.1.2 Internal low-speed RC clock RCL</u>	<u>37</u>
<u>3.1.3 External low-speed crystal oscillator clock XTL</u>	<u>38</u>
<u>3.1.4 External high-speed crystal oscillator clock XTH</u>	<u>38</u>
<u>3.1.5 Clock startup process.....</u>	<u>39</u>
<u>3.2 System clock switching.....</u>	<u>40</u>
<u>3.2.1 Standard clock switching process.....</u>	<u>40</u>
<u>3.2.2 Switch from RCH to XTL example.....</u>	<u>40</u>
<u>3.2.3 Example of switching from RCH to XTH.....</u>	<u>41</u>
<u>3.2.4 Switching from RCL to XTH example.....</u>	<u>42</u>
<u>3.2.5 Example of switching from RCH to RCL.....</u>	<u>43</u>
<u>3.2.6 Example of switching from RCL to RCH.....</u>	<u>43</u>
<u>3.2.7 RCH switching between different oscillation frequencies.....</u>	<u>44</u>
<u>3.3 Clock calibration module.....</u>	<u>45</u>
<u>3.4 Interrupt wake-up control.....</u>	<u>46</u>
<u>3.4.1 Method of executing interrupt service routine after waking up from deep sleep mode.....</u>	<u>46</u>
<u>3.4.2 The method of not executing interrupt service routine after waking up from deep sleep mode.....</u>	<u>46</u>
<u>3.4.3 Use exit hibernation feature.....</u>	<u>47</u>
<u>3.5 register.....</u>	<u>49</u>
<u>3.5.1 System Control Register 0 (SYSCTRL0)</u>	<u>50</u>
<u>3.5.2 System Control Register 1 (SYSCTRL1)</u>	<u>52</u>
<u>3.5.3 System Control Register 2 (SYSCTRL2)</u>	<u>54</u>
<u>3.5.4 RCH Control Register (RCH_CR)</u>	<u>55</u>
<u>3.5.5 Oscillation XTH Control Register (XTH_CR)</u>	<u>56</u>
<u>3.5.6 RCL Control Register (RCL_CR)</u>	<u>57</u>
<u>3.5.7 XTL Control Register (XTL_CR)</u>	<u>58</u>
<u>3.5.8 Peripheral module clock control register (PERI_CLKEN)</u>	<u>59</u>

<u>3.5.9 Systick clock control (SYSTICK_CR)</u>	<u>61</u>
<u>4 Reset Controller (RESET)</u>	<u>62</u>
<u>4.1 Introduction to reset controller.....</u>	<u>62</u>
<u>4.1.1 Power on and power off reset POR/BOR</u>	<u>63</u>
<u>4.1.2 External reset pin reset</u>	<u>63</u>
<u>4.1.3 WDT reset</u>	<u>63</u>
<u>4.1.4 PCA reset</u>	<u>63</u>
<u>4.1.5 LVD low voltage reset</u>	<u>63</u>
<u>4.1.6 Cortex-M0+ SYSRESETREQ reset</u>	<u>63</u>
<u>4.1.7 Cortex-M0+ LOCKUP reset</u>	<u>63</u>
<u>4.2 register.....</u>	<u>65</u>
<u>4.2.1 Reset Flag Register (Reset_flag)</u>	<u>65</u>
<u>4.2.2 Peripheral module reset control register (PERI_RESET)</u>	<u>66</u>
<u>5 Interrupt Controller (NVIC)</u>	<u>68</u>

5.1	Overview	68
5.2	Interrupt priority	68
5.3	Interrupt vector table	69
5.4	Interrupt input and suspend behavior	70
5.5	Interrupt wait	73
5.6	Interrupt source	73
5.7	Interrupt structure diagram	75
5.8	register	77
5.8.1	Interrupt enable setting register (SCS_SETENA)	77
5.8.2	Interrupt enable clear register (SCS_CLRENA)	78
5.8.3	Interrupt pending status setting register (SCS_SETPEND)	78
5.8.4	Interrupt pending status clear register (SCS_CLRPEND)	79
5.8.5	Interrupt Priority Register (SCS_IPR0)	80
5.8.6	Interrupt Priority Register (SCS_IPR1)	81
5.8.7	Interrupt Priority Register (SCS_IPR2)	82
5.8.8	Interrupt Priority Register (SCS_IPR3)	83
5.8.9	Interrupt Priority Register (SCS_IPR4)	84
5.8.10	Interrupt Priority Register (SCS_IPR5)	85
5.8.11	Interrupt Priority Register (SCS_IPR6)	86
5.8.12	Interrupt Priority Register (SCS_IPR7)	87
5.8.13	Interrupt mask special register (SCS_PRIMASK)	88
5.9	Basic software operation	89
5.9.1	External interrupt enable	89
5.9.2	NVIC interrupt enable and clear enable	89
5.9.3	NVIC Interrupt Pending and Clear Pending	89
5.9.4	NVIC interrupt priority	89
5.9.5	NVIC interrupt shielding	90
	6-port controller (GPIO)	91

Page 5

6.1	Introduction to Port Controller	91
6.2	Main features of port controller	92
6.3	Port controller function description	93
6.3.1	Port configuration function	93
6.3.2	Port writing	95
6.3.3	Port reading	96
6.3.4	Port multiplexing function	97
6.3.5	Port interrupt function	98
6.4	Port configuration operation	99
6.4.1	Port multiplexing operation process	99
6.4.2	Port interrupt operation flow	100
6.4.3	Port configuration operation flow	101
6.5	Port controller register description	102
6.5.1	Port P0	105
	6.5.1.1 Port P01 Function Configuration Register (P01_SEL)	105
	6.5.1.2 Port P02 Function Configuration Register (P02_SEL)	106
	6.5.1.3 Port P03 Function Configuration Register (P03_SEL)	107
	6.5.1.4 Port P0 Input and Output Configuration Register (P0DIR)	108
	6.5.1.5 Port P0 Input Value Register (P0IN)	109
	6.5.1.6 Port P0 output value configuration register (P0OUT)	110
	6.5.1.7 Port P0 Digital-to-Analog Configuration Register (P0ADS)	111
	6.5.1.8 Port P0 Drive Capability Configuration Register (P0DR)	112

6.5.1.9 Port P0 pull-up enable configuration register (P0PU)	113
6.5.1.10 Port P0 pull-down enable configuration register (P0PD)	114
6.5.1.11 Port P0 open-drain output configuration register (P0OD)	115
6.5.1.12 Port P0 high level interrupt enable configuration register (P0HIE)	116
6.5.1.13 Port P0 low-level interrupt enable configuration register (P0LIE)	117
6.5.1.14 Port P0 rising edge interrupt enable configuration register (P0RIE)	118
6.5.1.15 Port P0 Falling Edge Interrupt Enable Configuration Register (P0FIE)	119
6.5.1.16 Port P0 interrupt status register (P0_STAT)	120
6.5.1.17 Port P0 Interrupt Clear Register (P0_ICLR)	121
6.5.2 Port P1	122
6.5.2.1 Port P14 Function Configuration Register (P14_SEL)	122
6.5.2.2 Port P15 Function Configuration Register (P15_SEL)	123
6.5.2.3 Port P1 Input and Output Configuration Register (P1DIR)	124
6.5.2.4 Port P1 Input Value Register (P1IN)	125
6.5.2.5 Port P1 output value configuration register (P1OUT)	126
6.5.2.6 Port P1 Digital-to-Analog Configuration Register (P1ADS)	127
6.5.2.7 Port P1 Drive Capability Configuration Register (P1DR)	128
6.5.2.8 Port P1 pull-up enable configuration register (P1PU)	129
6.5.2.9 Port P1 pull-down enable configuration register (P1PD)	130
6.5.2.10 Port P1 open-drain output configuration register (P1OD)	131

Page 6

6.5.2.11 Port P1 High Level Interrupt Enable Configuration Register (P1HIE)	132
6.5.2.12 Port P1 low-level interrupt enable configuration register (P1LIE)	133
6.5.2.13 Port P1 rising edge interrupt enable configuration register (P1RIE)	134
6.5.2.14 Port P1 Falling Edge Interrupt Enable Configuration Register (P1FIE)	135
6.5.2.15 Port P1 Interrupt Status Register (P1_STAT)	136
6.5.2.16 Port P1 Interrupt Clear Register (P1_ICLR)	137
6.5.3 Port P2	138
6.5.3.1 Port P23 Function Configuration Register (P23_SEL)	138
6.5.3.2 Port P24 Function Configuration Register (P24_SEL)	139
6.5.3.3 Port P25 Function Configuration Register (P25_SEL)	140
6.5.3.4 Port P26 Function Configuration Register (P26_SEL)	141
6.5.3.5 Port P27 Function Configuration Register (P27_SEL)	142
6.5.3.6 Port P2 Input and Output Configuration Register (P2DIR)	143
6.5.3.7 Port P2 Input Value Register (P2IN)	144
6.5.3.8 Port P2 output value configuration register (P2OUT)	145
6.5.3.9 Port P2 Digital-to-Analog Configuration Register (P2ADS)	146
6.5.3.10 Port P2 Drive Capability Configuration Register (P2DR)	147
6.5.3.11 Port P2 pull-up enable configuration register (P2PU)	148
6.5.3.12 Port P2 pull-down enable configuration register (P2PD)	149
6.5.3.13 Port P2 open-drain output configuration register (P2OD)	150
6.5.3.14 Port P2 high level interrupt enable configuration register (P2HIE)	151
6.5.3.15 Port P2 low-level interrupt enable configuration register (P2LIE)	152
6.5.3.16 Port P2 rising edge interrupt enable configuration register (P2RIE)	153
6.5.3.17 Port P2 falling edge interrupt enable configuration register (P2FIE)	154
6.5.3.18 Port P2 Interrupt Status Register (P2_STAT)	155
6.5.3.19 Port P2 Interrupt Clear Register (P2_ICLR)	156
6.5.4 Port P3	157
6.5.4.1 Port P31 Function Configuration Register (P31_SEL)	157
6.5.4.2 Port P32 Function Configuration Register (P32_SEL)	158
6.5.4.3 Port P33 Function Configuration Register (P33_SEL)	159
6.5.4.4 Port P34 Function Configuration Register (P34_SEL)	160

6.5.4.5 Port P35 Function Configuration Register (P35_SEL)	161
6.5.4.6 Port P36 Function Configuration Register (P36_SEL)	162
6.5.4.7 Port P3 Input and Output Configuration Register (P3DIR)	163
6.5.4.8 Port P3 Input Value Register (P3IN)	165
6.5.4.9 Port P3 output value configuration register (P3OUT)	167
6.5.4.10 Port P3 Digital-to-Analog Configuration Register (P3ADS)	169
6.5.4.11 Port P3 Drive Capability Configuration Register (P3DR)	171
6.5.4.12 Port P3 pull-up enable configuration register (P3PU)	173
6.5.4.13 Port P3 pull-down enable configuration register (P3PD)	175
6.5.4.14 Port P3 open-drain output configuration register (P3OD)	177
6.5.4.15 Port P3 High Level Interrupt Enable Configuration Register (P3HIE)	179

Page 7

6.5.4.16 Port P3 low-level interrupt enable configuration register (P3LIE)	181
6.5.4.17 Port P3 rising edge interrupt enable configuration register (P3RIE)	183
6.5.4.18 Port P3 falling edge interrupt enable configuration register (P3FIE)	185
6.5.4.19 Port P3 interrupt status register (P3_STAT)	187
6.5.4.20 Port P3 Interrupt Clear Register (P3_ICLR)	189
6.5.5 Port auxiliary function	191
6.5.5.1 Port auxiliary function configuration register 1 (GPIO_CTRL1)	191
6.5.5.2 Port auxiliary function configuration register 2 (GPIO_CTRL2)	193
6.5.5.3 Port auxiliary function configuration register 3 (GPIO_CTRL3)	195
6.5.5.4 Port auxiliary function configuration register 4 (GPIO_CTRL4)	197
7 FLASH Controller (FLASH)	199
7.1 Overview	199
7.2 Structure diagram	199
7.3 Function description	200
7.3.1 Page Erase (Sector Erase)	200
7.3.2 Chip Erase	200
7.3.3 Write operation (Program)	201
7.3.4 Read operation	203
7.4 Erasing and writing sequence	204
7.5 Read wait period	206
7.6 Erase and write protection	206
7.6.1 Erase protection bit	206
7.6.2 PC address erasing and writing protection	206
7.7 Register write protection	207
7.8 register	208
7.8.1 TNVS parameter register (FLASH_TNVS)	208
7.8.2 TPGS parameter register (FLASH_TPGS)	209
7.8.3 TPROG parameter register (FLASH_TPROG)	209
7.8.4 TSERASF register (FLASH_TSERASE)	210
7.8.5 TMERASE parameter register (FLASH_TMERASE)	210
7.8.6 TPRCV parameter register (FLASH_TPRCV)	211
7.8.7 TSRCV parameter register (FLASH_TSRCV)	211
7.8.8 TMRCV parameter register (FLASH_TMRCV)	212
7.8.9 CR register (FLASH_CR)	212
7.8.10 IFR register (FLASH_IFR)	213
7.8.11 ICLR register (FLASH_ICLR)	213
7.8.12 BYPASS register (FLASH_BYPASS)	214
7.8.13 SLOCK register (FLASH_SLOCK)	215
8 RAM Controller (RAM)	216

8.1	Overview...	216
8.2	Function description...	216
8.3	register...	217

Page 8

8.3.1	Control Register (RAM_CR)	217
8.3.2	Parity error address register (RAM_ERRADDR)	218
8.3.3	Error interrupt flag register (RAM_IFR)	218
8.3.4	Error interrupt flag clear register (RAM_ICLR)	219
9	Basic timer (TIM0/1/2)	220
9.1	Introduction to basic timers...	220
9.2	Base Timer function description...	221
9.2.1	Counting function...	223
9.2.2	Timing function...	223
9.2.3	Buzzer function ...	224
9.3	Base Timer interconnection...	225
9.3.1	GATE interconnection...	225
9.3.2	Toggle Output Interconnection...	225
9.4	Base Timer register description...	226
9.4.1	16-bit mode reload register (TIMx_ARR)	226
9.4.2	16-bit mode count register (TIMx_CNT)	227
9.4.3	32-bit mode count register (TIMx_CNT32)	227
9.4.4	Control Register (TIMx_CR)	228
9.4.5	Interrupt Flag Register (TIMx_IFR)	229
9.4.6	Interrupt flag clear register (TIMx_ICLR)	229
10	Low Power Timer (LPTIM)	230
10.1	Introduction to LPTimer...	230
10.2	LPTimer function description...	231
10.2.1	Counting function...	232
10.2.2	Timing function...	232
10.3	LPTimer interconnection...	233
10.3.1	GATE interconnection...	233
10.3.2	EXT interconnection...	233
10.3.3	Toggle Output Interconnection...	233
10.4	LPTimer register description...	234
10.4.1	Counter count value register (LPTIM_CNT)	235
10.4.2	Reload register (LPTIM_ARR)	235
10.4.3	Control Register (LPTIM_CR)	236
10.4.4	Interrupt Flag Register (LPTIM_IFR)	237
10.4.5	Interrupt Flag Clear Register (LPTIM_ICLR)	237
11	Programmable Counting Array (PCA)	238
11.1	Introduction to PCA ...	238
11.2	PCA function description...	239
11.2.1	PCA Timer/Counter...	239
11.2.2	PCA capture function...	241
11.2.3	PCA comparison function...	243
	11.2.3.1 16-bit software counting mode...	243

<u>11.2.3.2 High-speed output mode</u>	244
<u>11.2.3.3 WDT function of PCA module 4</u>	245
<u>11.2.3.4 PCA 8-bit pulse width modulation function</u>	247
<u>11.3 PCA module and other modules interconnection and control</u>	249
<u>11.3.1 ECI interconnection</u>	249
<u>11.3.2 PCACAP0</u>	249
<u>11.3.3 PCACAP1</u>	249
<u>11.3.4 PCACAP [4:2]</u>	249
<u>11.4 PCA register description</u>	250
<u>11.4.1 Control Register (PCA_CCON)</u>	251
<u>11.4.2 Mode register (PCA_CMOD)</u>	252
<u>11.4.3 Counting register (PCA_CNT)</u>	253
<u>11.4.4 Interrupt Clear Register (PCA_ICLR)</u>	253
<u>11.4.5 Compare capture mode register (PCA_CCAPM0~4)</u>	254
<u>11.4.6 The upper 8 bits of the compare capture data register (PCA_CCAP0~4H)</u>	255
<u>11.4.7 Compare the lower 8 bits of the capture data register (PCA_CCAP0~4L)</u>	255
<u>11.4.8 Compare and capture 16-bit registers (PCA_CCAP0~4)</u>	256
<u>11.4.9 Compare high-speed output flag register (PCA_CCAPO)</u>	256
<u>12 Advanced Timer (TIM4/5/6)</u>	257
<u>12.1 Introduction to Advanced Timer</u>	257
<u>12.2 Advanced Timer function description</u>	259
<u>12.2.1 Basic actions</u>	259
<u>12.2.1.1 Basic Waveform Mode</u>	259
<u>12.2.1.2 Comparison output</u>	260
<u>12.2.1.3 Capture Input</u>	261
<u>12.2.2 Clock source selection</u>	262
<u>12.2.3 Counting direction</u>	263
<u>12.2.3.1 Sawtooth wave counting direction</u>	263
<u>12.2.3.2 Triangular wave counting direction</u>	263
<u>12.2.4 Digital Filtering</u>	264
<u>12.2.5 Software synchronization</u>	265
<u>12.2.5.1 Synchronous start of software</u>	265
<u>12.2.5.2 Software synchronization stop</u>	265
<u>12.2.5.3 Software Synchronous Clear</u>	265
<u>12.2.6 Hardware synchronization</u>	267
<u>12.2.6.1 Hardware synchronization start</u>	267
<u>12.2.6.2 Hardware synchronization stop</u>	267
<u>12.2.6.3 Hardware Synchronous Clear</u>	267
<u>12.2.6.4 Hardware synchronization capture input</u>	267
<u>12.2.6.5 Hardware synchronization counting</u>	267
<u>12.2.7 Cache function</u>	269
<u>12.2.7.1 Cache transmission time point</u>	270

<u>12.2.7.2 Common cycle reference value buffer transmission time point</u>	270
<u>12.2.7.3 General comparison reference value buffer transmission time point</u>	270
<u>12.2.7.4 Capture input value buffer transmission time point</u>	270
<u>12.2.7.5 Buffer transmission during clearing action</u>	270
<u>12.2.8 General purpose PWM output</u>	271

12.2.8.1 PWM spread spectrum output...	271
12.2.8.2 Independent PWM output...	271
12.2.8.3 Complementary PWM output...	271
 12.2.8.3.1 Software setting GCMBR complementary PWM output...	272
 12.2.8.3.2 Hardware setting GCMBR complementary PWM output...	273
12.2.8.4 Multi-phase PWM output...	274
12.2.9 Quadrature encoding count...	276
 12.2.9.1 Position counting mode...	276
 12.2.9.1.1 Basic counting...	276
 12.2.9.1.2 Phase difference counting...	277
 12.2.9.1.3 Direction count...	278
 12.2.9.2 Revolution mode...	278
 12.2.9.2.1 Z-phase counting...	278
 12.2.9.2.2 Position overflow counting...	279
 12.2.9.2.3 Mixed counting...	279
 12.2.9.2.4 Z-phase action shielding...	280
12.2.10 Periodic interval response...	282
12.2.11 protection mechanism...	283
12.2.12 Interrupt description...	284
 12.2.12.1 Counting compare match interrupt...	284
 12.2.12.2 Count period match interruption...	284
 12.2.12.3 Dead time error interrupt...	284
12.2.13 Brake protection...	285
 12.2.13.1 Port brake and software brake...	285
 12.2.13.2 Automatic brake in low power consumption mode...	285
 12.2.13.3 The output level is the same as the high and the same as the low brake...	286
 12.2.13.4 VC brake...	286
12.2.14 Internal interconnection...	287
 12.2.14.1 Interrupt trigger output...	287
 12.2.14.2 AOS Trigger...	287
 12.2.14.3 Port trigger TRIGA-TRIGD ...	288
 12.2.14.4 Interconnection of comparison output VC and Advanced Timer...	289
 12.2.14.5 UART and Advanced Timer interconnection...	289
12.3 Register description ...	290
12.3.1 General Counting Reference Value Register (TIMx_CNTER)...	292
12.3.2 General Period Reference Value Register (TIMx_PERAR)...	292
12.3.3 General Period Buffer Register (TIMx_PERBR) ...	293

12.3.4 General comparison reference value register (TIMx_GCMAR-GCMDR)...	293
12.3.5 Dead time reference value register (TIMx_DTUAR- DTDAR)	294
12.3.6 General Control Register (TIMx_GCONR)...	295
12.3.7 Interrupt Control Register (TIMx_ICONR) ...	297
12.3.8 Port Control Register (TIMx_PCONR)...	298
12.3.9 Buffer Control Register (TIMx_BCONR)...	301
12.3.10 Dead zone control register (TIMx_DCONR) ...	302
12.3.11 Filter control register (TIMx_FCONR)...	303
12.3.12 Effective Period Register (TIMx_VPERR) ...	305
12.3.13 Status Flag Register (TIMx_STFLR) ...	306
12.3.14 Hardware start event selection register (TIMx_HSTAR) ...	308
12.3.15 Hardware stop event selection register (TIMx_HSTPR) ...	310
12.3.16 Hardware clear event selection register (TIMx_HCFLR) ...	312

12.3.17	Hardware capture A event selection register (TIMx_HCPAR)	314
12.3.18	Hardware capture B event selection register (TIMx_HCPBR)	316
12.3.19	Hardware Increasing Event Selection Register (TIMx_HCUPR)	318
12.3.20	Hardware decrement event selection register (TIMx_HCDOR)	320
12.3.21	Software synchronization start register (TIMx_SSTAR)	322
12.3.22	Software synchronization stop register (TIMx_SSTPR)	323
12.3.23	Software Synchronous Clear Register (TIMx_SCLRR)	324
12.3.24	Interrupt Flag Register (TIMx_JFR)	325
12.3.25	Interrupt flag clear register (TIMx_ICLR)	327
12.3.26	Spread spectrum and interrupt trigger selection (TIMx_CR)	328
12.3.27	AOS selection control register (TIMx_AOSSR)	329
12.3.28	AOS selection control register flag clear (TIMx_AOSCL)	330
12.3.29	Port brake control register (TIMx_PTBKSI)	331
12.3.30	Port trigger control register (TIMx_TTRIG)	332
12.3.31	AOS trigger control register (TIMx_JTRIG)	333
12.3.32	Port brake polarity control register (TIMx_PTBKPO)	334
13 Real Time Clock (RTC)		335
13.1	Introduction to Real Time Clock	335
13.2	Real-time clock function description	336
13.2.1	Power-on setting	336
13.2.2	RTC counting start setting	336
13.2.3	System low-power mode switching	336
13.2.4	Read count register	337
13.2.5	Write count register	337
13.2.6	Alarm setting	338
13.2.7	1Hz output	338
13.2.8	Clock error compensation	339
13.3	RTC interrupt	341
13.3.1	RTC alarm interruption	341

Page 12

13.3.2	RTC Periodic Interruption	341
13.4	RTC register description	342
13.4.1	Control Register 0 (RTC_CR0)	343
13.4.2	Control Register 1 (RTC_CR1)	345
13.4.3	Second counter register (RTC_SEC)	347
13.4.4	Minute counter register (RTC_MIN)	347
13.4.5	Hour counter register (RTC_HOUR)	348
13.4.6	Day count register (RTC_DAY)	350
13.4.7	Week count register (RTC_WEEK)	351
13.4.8	Month count register (RTC_MON)	352
13.4.9	Year counting register (RTC_YEAR)	352
13.4.10	Sub-alarm register (RTC_ALMMIN)	353
13.4.11	Time alarm register (RTC_ALMHOUR)	353
13.4.12	Weekly alarm register (RTC_ALM WEEK)	354
13.4.13	Clock Error Compensation Register (RTC_COMPEN)	355
14 Watchdog Timer (WDT)		357
14.1	Introduction to WDT	357
14.2	WDT function description	358
14.2.1	An interrupt is generated after WDT overflows	358
14.2.2	Reset after WDT overflow	358
14.3	WDT register description	359
14.3.1	WDT Clear Control Register (WDT_RST)	359

<u>14.3.2</u>	<u>WDT_CON register.....</u>	360
<u>15 Universal Synchronous Asynchronous Receiver Transmitter (UART).....</u>		361
<u>15.1</u>	<u>Overview.....</u>	361
<u>15.2</u>	<u>Structure diagram.....</u>	361
<u>15.3</u>	<u>Main features.....</u>	362
<u>15.4</u>	<u>Function description.....</u>	363
<u>15.4.1</u>	<u>Operating mode.....</u>	363
<u>15.4.1.1</u>	<u>Mode0~Mode3 function comparison.....</u>	363
<u>15.4.1.2</u>	<u>Mode0 (synchronous mode, half-duplex).....</u>	363
<u>15.4.1.3</u>	<u>Mode1 (asynchronous mode, full duplex).....</u>	365
<u>15.4.1.4</u>	<u>Mode2 (asynchronous mode, full duplex).....</u>	365
<u>15.4.1.5</u>	<u>Mode3 (asynchronous mode, full duplex).....</u>	366
<u>15.4.2</u>	<u>Baud rate generation.....</u>	368
<u>15.4.2.1</u>	<u>Mode1/Mode3 baud rate setting example.....</u>	369
<u>15.5</u>	<u>Frame error detection.....</u>	373
<u>15.6</u>	<u>Multi-machine communication.....</u>	374
<u>15.7</u>	<u>Automatic address recognition.....</u>	375
<u>15.7.1</u>	<u>Given address.....</u>	375
<u>15.7.1.1</u>	<u>Broadcast address.....</u>	375
<u>15.7.1.2</u>	<u>Examples.....</u>	375

Page 13

<u>15.8</u>	<u>Transceiver side buffer.....</u>	376
<u>15.8.1</u>	<u>Receiving buffer.....</u>	376
<u>15.8.2</u>	<u>Sending buffer.....</u>	376
<u>15.9</u>	<u>register.....</u>	377
<u>15.9.1</u>	<u>Data register (UARTx_SBUF).....</u>	377
<u>15.9.2</u>	<u>Control Register (UARTx_SCON).....</u>	378
<u>15.9.3</u>	<u>Address register (UARTx_SADDR).....</u>	379
<u>15.9.4</u>	<u>Address mask register (UARTx_SADEN).....</u>	379
<u>15.9.5</u>	<u>Flag bit register (UARTx_ISR).....</u>	380
<u>15.9.6</u>	<u>Flag Clear Register (UARTx_ICR).....</u>	381
<u>16 Low Power Synchronous Asynchronous Receiver Transmitter (LPUART).....</u>		382
<u>16.1</u>	<u>Overview.....</u>	382
<u>16.2</u>	<u>Structure diagram.....</u>	383
<u>16.3</u>	<u>Main features.....</u>	384
<u>16.4</u>	<u>Function description.....</u>	385
<u>16.4.1</u>	<u>Configure clock and transmission clock.....</u>	385
<u>16.4.2</u>	<u>Operating mode.....</u>	385
<u>16.4.2.1</u>	<u>Mode0~Mode3 function comparison.....</u>	386
<u>16.4.2.2</u>	<u>Mode0 (synchronous mode, half-duplex) data transmission and reception description.....</u>	387
<u>16.4.2.3</u>	<u>Mode1 (asynchronous mode, full-duplex) data transmission and reception description.....</u>	388
<u>16.4.2.4</u>	<u>Mode2 (asynchronous mode, full-duplex) data transmission and reception description.....</u>	389
<u>16.4.2.5</u>	<u>Mode3 (asynchronous mode, full-duplex) data transmission and reception description.....</u>	390
<u>16.4.3</u>	<u>Baud rate generation.....</u>	391
<u>16.5</u>	<u>Frame error detection.....</u>	391
<u>16.6</u>	<u>Multi-machine communication.....</u>	392
<u>16.7</u>	<u>Automatic address recognition.....</u>	393
<u>16.7.1</u>	<u>Given address.....</u>	393
<u>16.7.2</u>	<u>Broadcast address.....</u>	393
<u>16.7.3</u>	<u>Example.....</u>	393
<u>16.8</u>	<u>Transceiver side buffer.....</u>	394

16.8.1	Receiving buffer	394
16.8.2	Sending buffer	394
16.9	register	395
16.9.1	Data register (LPUART_SBUF)	395
16.9.2	Control Register (LPUART_SCON)	396
16.9.3	Address register (LPUART_SADDR)	398
16.9.4	Address mask register (LPUART_SADEN)	398
16.9.5	Interrupt Flag Register (LPUART_ISR)	399
16.9.6	Interrupt flag bit clear register (LPUART_ICR)	400
17	I2C Bus (I2C)	401
17.1	Introduction	401
17.2	Main features	401

HC32L110 Series User Manual Rev2.31 Page 13 of 527

Page 14

17.3	Protocol description	401
17.3.1	Data transmission on I2C bus	402
17.3.2	Response on I2C bus	403
17.3.3	Arbitration on the I2C bus	404
17.4	Function description	406
17.4.1	Serial clock generator	407
17.4.2	Input filter	407
17.4.3	Address Comparator	407
17.4.4	Response flag	408
17.4.5	Interrupt generator	408
17.4.6	Operating mode	408
17.4.7	Status code description	415
17.5	Programming example	417
17.5.1	Host sending example	417
17.5.2	Host reception example	418
17.5.3	Example of receiving from the machine	419
17.5.4	Example of slave sending	420
17.6	Register description	421
17.6.1	I2C baud rate counter enable register (I2C_TMRUN)	421
17.6.2	I2C baud rate counter configuration register (I2C_TM)	422
17.6.3	I2C Configuration Register (I2C_CR)	423
17.6.4	I2C Data Register (I2C_DATA)	425
17.6.5	I2C address register (I2C_ADDR)	426
17.6.6	I2C Status Register (I2C_STAT)	427
18	Serial Peripheral Interface (SPI)	428
18.1	Introduction to SPI	428
18.2	SPI main features	428
18.3	SPI function description	429
18.3.1	SPI master mode	429
18.3.2	SPI slave mode	429
18.3.3	SPI data frame format	431
18.3.4	SPI status flags and interrupts	432
18.3.5	SPI multi-machine system configuration instructions	432
18.3.6	SPI pin configuration description	434
18.4	SPI programming example	435
18.4.1	SPI master sending example	435
18.4.2	SPI host receiving example	435
18.4.3	SPI slave sending example	436

18.6	18.4.4 SPI slave receiving example.....	436
18.6.1	SPI Configuration Register (SPI_CR).....	439
18.6.2	SPI Chip Select Configuration Register (SPI_SSN).....	441

Page 15

18.6.3	SPI Status Register (SPI_STAT).....	442
18.6.4	SPI Data Register (SPI_DATA).....	443
19 Clock Calibration Module (CLKTRIM).....		444
19.1	Introduction to CLK_TRIM	444
19.2	CLK_TRIM main features.....	444
19.3	CLK_TRIM function description.....	445
19.3.1	CLK_TRIM calibration mode.....	445
19.3.1.1	Operation process.....	445
19.3.2	CLK_TRIM monitoring mode.....	446
19.3.2.1	Operation process.....	446
19.4	CLK_TRIM register description.....	447
19.4.1	Configuration Register (CLKTRIM_CR)	448
19.4.2	Reference counter initial value configuration register (CLKTRIM_REFCON)	449
19.4.3	Reference counter value register (CLKTRIM_REFCNT)	449
19.4.4	Calibration counter value register (CLKTRIM_CALCNT).....	450
19.4.5	Interrupt Flag Bit Register (CLKTRIM_IFR)	451
19.4.6	Interrupt flag bit clear register (CLKTRIM_ICLR)	452
19.4.7	Calibration counter overflow value configuration register (CLKTRIM_CALCON)	453
20 Cyclic Redundancy Check (CRC).....		454
20.1	Overview.....	454
20.2	Main features.....	454
20.3	Function description.....	454
20.3.1	Operating mode.....	454
20.3.2	Encoding.....	454
20.3.3	Write bit width.....	454
20.4	Programming example.....	455
20.4.1	CRC-16 coding mode.....	455
20.4.2	CRC-16 check mode.....	455
20.5	Register description	456
20.5.1	Register list.....	456
20.5.2	Result register (CRC_RESULT)	457
20.5.3	Data register (CRC_DATA)	457
21 Analog-to-digital converter (ADC).....		458
21.1	Module introduction.....	458
21.2	ADC block diagram.....	458
21.3	Conversion timing and conversion speed.....	459
21.4	Single conversion mode.....	460
21.5	Continuous conversion mode.....	462
21.6	Continuous conversion accumulation mode.....	464
21.7	ADC conversion result comparison.....	466
21.8	ADC interrupt	467
21.9	Use temperature sensor to measure ambient temperature.....	467

<u>21.10 ADC module register.....</u>	469
<u>21.10.1 ADC configuration register 0 (ADC_CR0).....</u>	470
<u>21.10.2 ADC Configuration Register 1 (ADC_CR1).....</u>	472
<u>21.10.3 ADC Configuration Register 2 (ADC_CR2).....</u>	475
<u>21.10.4 ADC channel 0 conversion result (ADC_result0).....</u>	477
<u>21.10.5 ADC channel 1 conversion result (ADC_result1).....</u>	477
<u>21.10.6 ADC channel 2 conversion result (ADC_result2).....</u>	478
<u>21.10.7 ADC channel 3 conversion result (ADC_result3).....</u>	478
<u>21.10.8 ADC channel 4 conversion result (ADC_result4).....</u>	479
<u>21.10.9 ADC channel 5 conversion result (ADC_result5).....</u>	479
<u>21.10.10 ADC channel 6 conversion result (ADC_result6).....</u>	480
<u>21.10.11 ADC channel 7 conversion result (ADC_result7).....</u>	480
<u>21.10.12 ADC channel 8 conversion result (ADC_result8).....</u>	481
<u>21.10.13 ADC conversion result accumulated value (ADC_result_acc).....</u>	481
<u>21.10.14 ADC comparison upper threshold (ADC_HT).....</u>	482
<u>21.10.15 ADC comparison lower threshold (ADC_LT).....</u>	482
<u>21.10.16 ADC Interrupt Flag Register (ADC_IFR).....</u>	483
<u>21.10.17 ADC Interrupt Clear Register (ADC_ICLR).....</u>	484
<u>21.10.18 ADC result (ADC_result).....</u>	484
<u>22 Analog Comparator (VC).....</u>	485
<u>22.1 Introduction to Analog Voltage Comparator VC.....</u>	485
<u>22.2 Frame Diagram of Voltage Comparator.....</u>	486
<u>22.3 Build/response time.....</u>	486
<u>22.4 Filtering time.....</u>	487
<u>22.5 Hysteresis function.....</u>	487
<u>22.6 VC register.....</u>	488
<u>22.6.1 VC Configuration Register (VC_CR).....</u>	489
<u>22.6.2 VC0 Configuration Register (VC0_CR).....</u>	491
<u>22.6.3 VC1 Configuration Register (VC1_CR).....</u>	493
<u>22.6.4 VC0 output configuration register (VC0_OUT_CFG).....</u>	495
<u>22.6.5 VC1 output configuration register (VC1_OUT_CFG).....</u>	497
<u>22.6.6 VC Interrupt Register (VC_IFR).....</u>	499
<u>23 Low Voltage Detector (LVD).....</u>	500
<u>23.1 Introduction to LVD.....</u>	500
<u>23.2 LVD block diagram.....</u>	500
<u>23.3 Digital Filtering.....</u>	501
<u>23.4 Hysteresis function.....</u>	501
<u>23.5 Configuration example.....</u>	502
<u>23.5.1 LVD is configured as low voltage reset.....</u>	502
<u>23.5.2 LVD is configured as voltage change interruption.....</u>	502
<u>23.6 LVD register.....</u>	503
<u>23.6.1 LVD Configuration Register (LVD_CR).....</u>	503

<u>23.6.2 LVD Interrupt Register (LVD_IFR).....</u>	505
<u>24 Simulate other registers.....</u>	506
<u>24.1 BGR Configuration Register (BGR_CR).....</u>	506
<u>25 SWD debug interface.....</u>	507

25.1	SWD debugging additional function.....	507
25.2	ARM® Reference Documents.....	508
25.3	Debug port pins.....	509
25.3.1	SWD port pins.....	509
25.3.2	SW-DP pin assignment.....	509
25.3.3	Internal pull-up on the SWD pin.....	509
25.4	SWD port.....	510
25.4.1	Introduction to SWD Protocol.....	510
25.4.2	SWD protocol sequence.....	510
25.4.3	SW-DP state machine (reset, idle state, ID code).....	511
25.4.4	DP and AP read/write access.....	511
25.4.5	SW-DP register.....	512
25.4.6	SW-AP register.....	513
25.5	Kernel debugging.....	514
25.6	BPU (Break Point Unit).....	514
25.6.1	BPU function	514
25.7	DWT (Data Observation Point).....	515
25.7.1	DWT function	515
25.7.2	DWT Program Counter Sampling Register.....	515
25.8	MCU debug component (DBG).....	516
25.8.1	Debugging support for low power consumption mode.....	516
25.8.2	Debugging support for timers and watchdogs.....	516
25.9	Debug mode module working status control (DEBUG_ACTIVE).....	517
26	Device electronic signature.....	519
26.1	Product unique identification (UID) register (80 bits).....	519
26.2	Product model register.....	520
26.3	FLASH capacity register.....	520
26.4	RAM capacity register.....	521
26.5	Number of pins register.....	521
27	Appendix A SysTick Timer.....	522
27.1	Introduction to SysTick Timer.....	522
27.2	Setting up SysTick	522
27.3	SysTick register	523
27.3.1	SysTick Control and Status Register (CTRL)	523
27.3.2	SysTick reload register (LOAD)	523
27.3.3	SysTick current value register (VAL)	524
28	Appendix B Document Conventions.....	525
28.1	List of register-related abbreviations.....	525

28.2	Glossary.....	525
	Version history & contact information.....	526

Page 19

Table catalog

Table 2-1 Diagram of modules that can be run in operation mode	30
Table 2-2 Diagram of modules that can be run in sleep mode	32
Table 2-3 Diagram of modules that can be run in deep sleep mode	34
Table 5-1 Cortex-M0+ processor exception list	68
Table 6-2 Correspondence between external interrupt and NVIC interrupt input	74
Table 6-1 Truth Table of Port Status	94
Table 6-2 Port reuse table	97
Table 7-1 FLASH erasing time parameters under different frequencies	204
Table 9-1 List of Base Timer Registers	226
Table 10-1 LPTimer register list	234
Table 11-1 PCA comparison capture function module settings	248
Table 11-2 PCA register list	250
Table 12-1 Basic Features of Advanced Timer	257
Table 12-2 Advanced Timer port list	257
Table 12-3 AOS source selection	288
Table 12-4 Port trigger selection	288
Table 12-5 Advanced Timer register list	291
Table 13-1 Basic characteristics of RTC	335
Table 13-2 List of RTC Registers	342

Table 14-1 List of WDT registers...	359
Table 15-1 Mode0/1/2/3 Data Structure...	363
Table 16-1 Mode0/1/2/3 Data Structure...	386
Table 17-1 I2C clock signal baud rate...	407
Table 17-2 I2C status code description...	416
Table 17-3 Register List...	421
Table 18-1 SPI pin configuration description table...	434
Table 18-2 SPI register list...	438
Table 18-3 Host mode baud rate selection...	440
Table 19-1 Register list...	447
Table 21-1 ADC Registers...	469
Table 22-1 VC Register...	488
Table 23-1 LVD Register...	503

List of Figures

Figure 1-1 Schematic diagram of system architecture ...	twenty four
Figure 1-2 Schematic diagram of address area division...	26
Figure 2-1 Block diagram of control mode ...	28
Figure 3-1 Block diagram of the clock control module...	37
Figure 3-2 Schematic diagram of crystal oscillator clock start...	39
Figure 3-3 Schematic diagram of clock switching ...	42
Figure 3-4 Schematic diagram of clock calibration...	45
Figure 4-1 Schematic diagram of reset source...	62
Figure 5-1 Only the upper two bits of the priority register are used...	68
Figure 5-2 Interrupt vector table ...	69
Figure 5-3 Interrupt activation and suspension status...	70
Figure 5-4 The interrupt pending state is cleared and then reconfirmed...	71
Figure 5-5 If the interrupt request remains high when the interrupt exits, it will cause the interrupt processing to be executed again...	71
Figure 5-6 The pending interrupt generated during interrupt processing can also be confirmed...	72
Figure 5-7 Interrupt structure diagram...	75
Figure 6-1 Schematic diagram of port circuit...	95
Figure 6-2 Changes of AHB bus port with system clock...	95
Figure 6-3 Read port pin data synchronization diagram...	96
Figure 7-1 Division of memory sectors...	199
Figure 9-1 Block diagram of Base Timer...	220
Figure 9-2 Block diagram of Timer Mode 1...	222
Figure 9-3 Block diagram of Timer Mode 2...	222
Figure 9-4 Timing diagram of Mode 1...	223
Figure 9-5 Timing diagram of Mode 2 (prescaler is set to 2)	223
Figure 10-1 LPTimer block diagram...	230
Figure 10-2 LPTimer Mode 1 ...	231
Figure 10-3 LPTimer Mode 2 ...	232

Figure 11-1 Overall block diagram of PCA...	238
Figure 11-2 Block diagram of PCA counter...	240
Figure 11-3 PCA capture function block diagram...	242
Figure 11-4 PCA comparison function block diagram...	244
Figure 11-5 PCA WDT functional block diagram...	245
Figure 11-6 PCA PWM functional block diagram...	247
Figure 11-7 PCA PWM output waveform...	248
Figure 12-1 Block diagram of Advanced Timer...	258
Figure 12-2 Sawtooth waveform (increasing counting)	259
Figure 12-3 Triangle wave waveform...	259
Figure 12-4 Compare output action...	260
Figure 12-5 Capture input action...	261
Figure 12-6 The filter function of the capture input port...	264

Page 21

Figure 12-7 Software synchronization action...	265
Figure 12-8 Hardware synchronization action...	268
Figure 12-9 Comparison output timing of single buffer mode...	269
Figure 12-10 Schematic diagram of PWM spread spectrum output...	271
Figure 12-11 CHA output PWM wave...	271
Figure 12-12 Software setting of GCMBR complementary PWM wave output in triangle wave A mode...	272
Figure 12-13 Hardware setting of GCMBR complementary PWM wave output in triangle wave B mode (symmetrical dead zone)	273
Figure 12-14 6-phase PWM wave...	274
Figure 12-15 Three-phase complementary PWM wave output with dead time in triangle wave A mode...	275
Figure 12-16 Basic counting action in position mode...	276
Figure 12-17 Phase difference counting action setting in position mode (1 times)	277
Figure 12-18 Phase difference counting action setting in position mode (2 times)	277
Figure 12-19 Phase difference counting action setting in position mode (4 times)	277
Figure 12-20 Direction counting action in position mode...	278
Figure 12-21 Phase Z counting action in revolution mode...	278
Figure 12-22 Position counter output counting action in revolution mode...	279
Figure 12-23 Mixed counting action of Z-phase counting and position counter output in revolution mode...	279
Figure 12-24 Revolution counting mode-mixed counting Z phase shielding action example 1	280
Figure 12-25 Revolution counting mode-mixed counting Z phase shielding action example 2	281
Figure 12-26 Periodic interval valid request signal action...	282
Figure 12-27 Schematic diagram of port brake and software brake...	285
Figure 12-28 Schematic diagram of output with the same high and low brake...	286
Figure 12-29 Schematic diagram of VC brake control...	286
Figure 12-30 Timer4/5/6 interrupt selection...	287
Figure 13-1 RTC block diagram...	335
Figure 14-1 Overall block diagram of WDT...	357
Figure 15-1 UART structure block diagram...	361
Figure 15-2 Mode0 sending data...	364
Figure 15-3 Mode0 receiving data...	364
Figure 15-4 Mode1 sending data...	365
Figure 15-5 Mode1 receiving data...	365
Figure 15-6 Mode2 sending data...	366
Figure 15-7 Mode2 receiving data...	366
Figure 16-1 LPUART structure block diagram...	383
Figure 16-2 Mode0 sending data...	387
Figure 16-3 Mode0 receiving data...	387
Figure 16-4 Mode1 sending data...	388
Figure 16-5 Mode1 receiving data...	388

Figure 16-6 Mode2 sending data...	389
Figure 16-7 Mode2 receiving data...	389
Figure 17-1 I2C transmission protocol	402
Figure 17-2 START and STOP conditions...	402

Page 22

Figure 17-3 I2C bus upper transmission...	403
Figure 17-4 Acknowledge signal on I2C bus	404
Figure 17-5 Arbitration on the I2C bus...	405
Figure 17-6 I2C function module diagram	406
Figure 17-7 Data synchronization diagram of master sending mode	409
Figure 17-8 I2C host sending state diagram	409
Figure 17-9 Data synchronization diagram of master receiving mode	410
Figure 17-10 I2C host receiving state diagram	411
Figure 17-11 Data synchronization diagram in slave receiving mode	412
Figure 17-12 Slave receiving state diagram	412
Figure 17-13 Data synchronization diagram in slave sending mode	413
Figure 17-14 I2C slave sending status diagram	413
Figure 17-15 I2C broadcast call state diagram	414
Figure 18-1 Schematic diagram of slave receiving	430
Figure 18-2 Schematic diagram of slave sending	430
Figure 18-3 Host mode frame format	431
Figure 18-4 Data frame format when slave CPHA is 0	431
Figure 18-5 Data frame format when the slave CHPA is 1	432
Figure 18-6 Schematic diagram of SPI multi-master/multi-slave system	433
Figure 21-1 Schematic block diagram of ADC	458
Figure 21-2 ADC conversion timing diagram	459
Figure 21-3 Example of ADC continuous conversion process	462
Figure 21-4 ADC continuous conversion and accumulation process example	464
Figure 22-1 VC frame diagram	486
Figure 22-2 VC filter response time	487
Figure 22-3 VC hysteresis function	487
Figure 23-1 LVD block diagram	500
Figure 23-2 LVD filter output	501
Figure 23-3 LVD hysteresis response	501

Introduction

The HC32L110 series is an ultra-low power consumption, Low Pin Count, MCU with wide voltage working range. Integrated 12-bit 1Msps high-precision SARADC and integrated comparator, multiple channels Rich communication peripherals such as UART, SPI, I2C, etc., have the characteristics of high integration, high anti-interference, high reliability and ultra-low p The core of this product adopts the Cortex-M0+ core, cooperates with mature Keil & IAR debugging and development software, supports C language and sinks Programming language, assembly instructions.

Typical application of ultra-low power MCU

- Sensor application, IoT application
- Smart transportation, smart city, smart home
- Fire alarm probe, smart door lock, wireless monitoring and other smart sensor applications
- A variety of portable devices that require battery power and power consumption, etc.

About this manual

This manual mainly introduces the function, operation items and usage method of the chip. For chip specifications, please refer to the corresponding "Data Manu book".

1 system structure

1.1 Overview

This product system consists of the following parts:

- 1 AHB bus Master:

-Cortex-M0+

- 4 AHB bus Slaves:

-FLASH memory

-SRAM memory

-AHB0, AHB to APB Bridge, including all APB interface peripherals

-AHB1, including all AHB interface peripherals

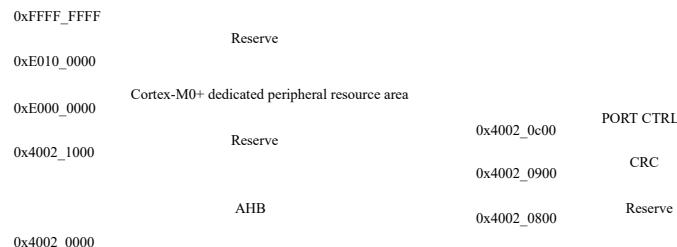
The bus structure of the whole system is realized by multi-level AHB-lite bus interconnection. As shown below:



Figure 1-1 Schematic diagram of system architecture

1.2 System address division

The address area division of the entire HC32L110 system is shown in the following figure:



0x4002_0400	
0x4002_0000	flash CTRL

	0x4000_3C00	Reserve
	0x4000_3800	TIM6
Reserve	0x4000_3400	TIM5
	0x4000_3000	TIM4
	0x4000_2C00	Reserve
	0x4000_2800	Reserve
	0x4000_2400	analog_ctrl
	0x4000_2000	System_ctrl
0x4000_4000	0x4000_1C00	Reserve
0x4000_0000	0x4000_1800	CLKTRIM
0x2000_1000	0x4000_1400	RTC
SRAM (Max 4KByte)	0x4000_1000	PCA
0x2000_0000	0x4000_0C00	TIM
0x0000_8000	0x4000_0800	SPI
Main flash area (Maximum 32KByte)	0x4000_0400	I2C
0x0000_0000	0x4000_0000	UART

Page 26

HC32L110C6UA	HC32L110C4UA
HC32L110C6PA	HC32L110C4PA
HC32L110B6PA	HC32L110B4PA
HC32L110B6YA	

Reserve	
0x2000_1000	Reserve

SRAM (4KByte)	0x2000_0800
0x2000_0000	SRAM (2KByte)
	0x2000_0000

0x0000_8000

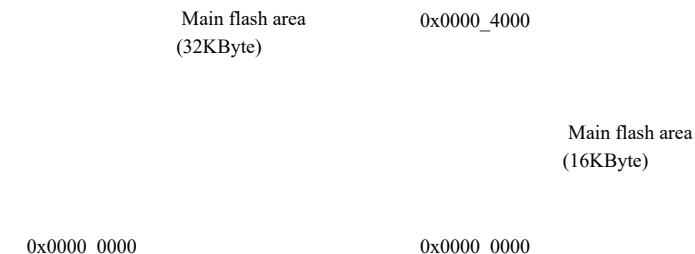


Figure 1-2 Schematic diagram of address area division

1.3 Memory and module address allocation

Boundary Address	Size	Memory Area	Description
0x0000_0000 – 0x0000_7FFF	32kByte	FLASH Memory	
0x0000_8000 – 0x0010_0BFF	-	Reserved	
0x0010_0C00 – 0x0010_0C3B	60Byte	Trim Data	
0x0010_0C3C-0x0010_0E6F-	-	Reserved	
0x0010_0E70 – 0x0010_0E7F	16Byte	UID	
0x0010_0E80-0x1FFF_FFFF-	-	Reserved	
0x2000_0000 – 0x2000_0FFF	4kByte	SRAM Memory	
0x2000_1000 – 0x3FFF_FFFF-	-	Reserved	
0x4000_0000 – 0x4000_0OFF	256Byte	UART0	
0x4000_0100 – 0x4000_01FF	256Byte	UART1	
0x4000_0200 – 0x4000_02FF	256Byte	LPUART	
0x4000_0300 – 0x4000_03FF	-	Reserved	
0x4000_0400 – 0x4000_07FF	1kByte	I2C	
0x4000_0800 – 0x4000_0BFF	1kByte	SPI	
0x4000_0C00 – 0x4000_0FFF	1kByte	Timer0/1/2/WDT/LPTimer	
0x4000_1000 – 0x4000_13FF	1kByte	PCA	
0x4000_1400 – 0x4000_17FF	1kByte	RTC	
0x4000_1800 – 0x4000_1BFF	1kByte	CLKTRIM	
0x4000_1C00 – 0x4000_1FFF-	-	Reserved	
0x4000_2000 – 0x4000_23FF	1kByte	SYSTEMCTRL	
0x4000_2400 – 0x4000_27FF	1kByte	ANALOGCTRL	
0x4000_2800 – 0x4000_2FFF	-	Reserved	
0x4000_3000 – 0x4000_33FF	1kByte	Timer4	
0x4000_3400 – 0x4000_37FF	1kByte	Timer5	

0x4000_3800 – 0x4000_3BFF	1kByte	Timer6
0x4000_3C00-0x4001_FFFF	-	Reserved
0x4002_0000-0x4002_03FF	1kByte	FLASH CTRL
0x4002_0400-0x4002_07FF	1kByte	RAM CTRL
0x4002_0800-0x4002_08FF	256Byte	Reserved
0x4002_0900-0x4002_0BFF	768Byte	CRC
0x4002_0C00-0x4002_0FFF	1kByte	PORT CTRL

Page 28

2 working mode

The power management module of this product is responsible for managing the switching between various working modes of this product and controlling the working status of each functional module under. The working voltage (VCC) of this product is 1.8 ~ 5.5V.

This product has the following working modes:

- (1) Operation mode: CPU operation, peripheral function module operation.
- (2) Sleep mode: CPU stops running and peripheral function modules are running.
- (3) Deep sleep mode: CPU stops running and high-speed clock stops running.

From the running mode, by executing the software program, you can enter other low power consumption modes. From various other low-power modes, the interrupt trigger can return to the running mode.

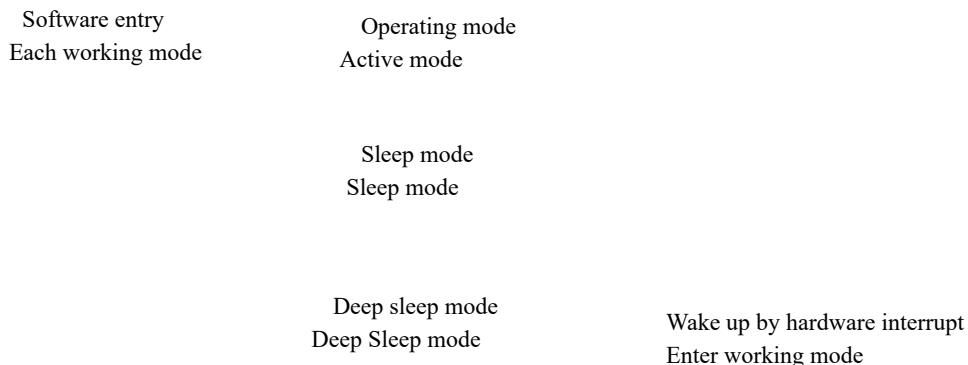


Figure 2-1 Block diagram of control mode

In each mode, the CPU can respond to all interrupt types.

	Interrupt source	Operating mode	Sleep mode	Deep sleep mode
[0]	GPIO_P0	✓	✓	✓
[1]	GPIO_P1	✓	✓	✓
[2]	GPIO_P2	✓	✓	✓
[3]	GPIO_P3	✓	✓	✓
[6]	UART0	✓	✓	

[7]	UART1	✓	✓	
[8]	LPUART	✓	✓	✓
[10]	SPI	✓	✓	
[12]	I2C	✓	✓	

Page 29

[14]	Timer0	✓	✓	
[15]	Timer1	✓	✓	
[16]	Timer2	✓	✓	
[17]	LPTimer	✓	✓	✓
[18]	Timer4	✓	✓	
[19]	Timer5	✓	✓	
[20]	Timer6	✓	✓	
[twenty one]	PCA	✓	✓	
[twenty two]	WDT	✓	✓	✓
[twenty three]	RTC	✓	✓	✓
[twenty four]	ADC	✓	✓	
[26]	VC0	✓	✓	✓
[27]	VC1	✓	✓	✓
[28]	LVD	✓	✓	✓
[30]	RAM FLASH	✓	✓	
[31]	CLKTRIM	✓	✓	✓

In each mode, this product can respond to all reset types.

	Reset source	Operating mode	Sleep mode	Deep sleep mode
[0]	Power-on and power-down reset POR		✓	✓
[1]	External Reset Pin reset	✓	✓	✓
[2]	LVD reset	✓	✓	✓
[3]	WDT reset	✓	✓	✓
[4]	PCA reset	✓	✓	
[5]	Cortex-M0+ LOCKUP	✓		
	Hardware reset			
[6]	Cortex-M0+	✓		
	SYSRESETREQ software			
	Reset			

2.1 Operating mode

The operating mode of this product (Active Mode):

After the system is reset after power-on, or wakes up from various low power consumption, the microcontroller MCU is in the running state. When the CP

When you don't need to continue running, you can use a variety of low-power modes to save energy, such as waiting for an external event. User needs

According to the lowest energy consumption, fastest startup time, available wake-up source and other conditions, select an optimal low-power mode.

Active Mode		
Cortex-M0+	SWD	XTH
FLASH	UART0-1	RCH
RAM	SPI	ADC
TIM0-2	I2C	RESET
TIM4-6	CRC	POR/BOR
PCA	XTL	LVD
LPUART	RCL	VC
LPTIM	RTC	CLKTRIM
GPIO	WDT OSC	WDT

Table 2-1 Diagram of modules that can be run in operation mode

Several methods to reduce chip power consumption in operating mode:

1) In the running mode, through the prescaler register (SYSCLK0.AHB_CLK_DIV,

SYSCLK0.APB_CLK_DIV) for programming, you can reduce any system clock (HCLK, PCLK)

speed. Before entering sleep mode, the prescaler can also be used to reduce the peripheral clock.

2) In the run mode, turn off the clocks (PERI_CLKx) that do not use peripherals to reduce power consumption.

3) In the run mode, turn off the clocks (PERI_CLKx) that do not use peripherals to reduce power consumption and allow the system to enter suspend

Reduce power consumption even more in sleep mode, and turn off clocks that do not use peripherals before executing WFI instructions

(PERI_CLKx).

4) Use low-power mode instead of sleep mode, because the wake-up time of this product is very short (~4uS), which can also meet the needs of the system

The demand of real-time response.

2.2 Sleep mode

This product sleep mode (Sleep Mode)

The WFI instruction can be used to enter the sleep mode. In the sleep mode, the CPU stops running, but the clock module, the system time

Clock, NVIC interrupt processing and peripheral function modules can still work.

The system enters the dormant state and does not change the port state. Before entering the dormant state, the IO state is changed to dormant state as needed. The system status.

How to enter sleep mode:

Enter the sleep state by executing the WFI instruction. According to the Cortex™ -M0+ system control register

With the value of the SLEEPONEXIT bit, there are two options for selecting the sleep mode entry mechanism:

SLEEP-NOW: If the SLEEPONEXIT bit is cleared, when WFI or WFE is executed, the microcontroller

Enter sleep mode immediately.

SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the system starts from the lowest priority interrupt handler

When exiting, the microcontroller immediately enters sleep mode.

How to exit sleep mode:

If the WFI instruction is executed to enter the sleep mode, any high-priority nested vectored interrupt controller responds to the peripheral

All interrupts can wake up the system from sleep mode.

Use note:

- SLEEP-ON-EXIT This bit is set to 1, it will automatically enter sleep after the interrupt is executed, and the program does not need to write __wfi();

- SLEEP-ON-EXIT This bit is cleared to 0, main() enters sleeping after __wfi() is executed, the interrupt is triggered and the execution is complete

After the interrupt program returns to main(), it enters sleeping after executing the WFI instruction. Wait for the subsequent interrupt to trigger.

- The SLEEP-ON-EXIT bit does not affect the execution of the __wfi() instruction. SLEEP-ON-EXIT =0: main() execution

After wfi() enters sleeping, the interrupt is triggered and the interrupt program returns to main() after executing the interrupt program, then continue to execute.

- If you enter sleep during an interrupt, only interrupts with a higher priority than this interrupt can be awakened, and the higher priority is executed first

Then execute low priority; interrupts whose priority is lower than or equal to this interrupt cannot be awakened.

Sleep Mode

Cortex-M0+	SWD	XTH
FLASH	UART0-1	RCH
RAM	SPI	ADC
TIM0-2	I2C	RESET
TIM4-6	CRC	POR/BOR
PCA	XTL	LVD
LPUART	RCL	VC
LPTIM	RTC	CLKTRIM
GPIO	WDT OSC	WDT

The gray modules do not work in the current state.

Table 2-2 Diagram of modules that can be run in sleep mode

2.3 Deep sleep mode

This product's deep sleep mode (Deep Sleep Mode)

Use SLEEPDEEP with WFI instruction to enter deep sleep mode. In deep sleep mode, the CPU stops

Stop running, the high-speed clock is turned off, the low-speed clock can be configured to run, and some low-power peripheral modules can be configured

Yes, NVIC interrupt processing can still work.

The system enters the deep sleep mode from the high-speed clock, the high-speed clock is automatically turned off, and the low-speed clock keeps entering

The previous state.

The system enters the deep sleep mode from the low-speed clock. Since the low-speed clock will not automatically shut down, it will keep running and

Sleep mode. Only the ARM Cortex-M0+ does not run, all other modules run.

When the system clock is switched, all clocks will not be turned off automatically, and the software needs to be turned off and turned on according to the

The corresponding clock.

The system enters the deep sleep state and will not change the port state. Before entering the sleep state, change the IO state as needed.

The state in hibernation. The unused IO pins and the IO pins not derived from the package need to be set as inputs and enabled

pull.

How to enter deep sleep mode:

First set the SLEEPDEEP bit in the Cortex-M0+ system control register, and enter by executing the WFI instruction

Sleep state. According to the value of the SLEEPONEXIT bit in the Cortex™-M0+ system control register, there are two options

Item can be used to select the deep sleep mode entry mechanism:

SLEEP-NOW: If the SLEEPONEXIT bit is cleared, when WFI or WFE is executed, the microcontroller

Enter sleep mode immediately.

SLEEP-ON-EXIT: If the SLEEPONEXIT bit is set, the system starts from the lowest priority interrupt handler

When exiting, the microcontroller immediately enters sleep mode.

How to exit deep sleep mode:

If the WFI instruction is executed to enter the sleep mode, any peripheral interrupt that is responded by the nested vector interrupt controller

(Interrupts of peripheral modules that can run under Deep Sleep) can wake up the system from sleep mode.

For wake-up settings, refer to [3.4 Interrupt wake-up control WIC](#).

Deep Sleep Mode

Cortex-M0+	SWD	XTH
FLASH	UART0-1	RCH
RAM	SPI	ADC
TIM0-2	I2C	RESET
TIM4-6	CRC	POR/BOR
PCA	XTL	LVD
LPUART	RCL	VC
LPTIM	RTC	CLKTRIM
GPIO	WDT OSC	WDT

The gray modules do not work in the current state.

Table 2-3 Diagram of modules that can be run in deep sleep mode

Page 35

System control register (Cortex-M0+ core system control register)

Address: 0xE000ED10

Reset value: 0x0000 0000

Bit	mark	Function description	Read and write
31:5	RESERVED	Reserve	
4	SEVONPEND	When set to 1, an event will be generated every time a new interrupt is suspended WFE hibernation is used, which can be used to wake up the processor	RW
3	RESERVED	Reserve	
2	SLEEPDEEP	When set to 1, execute WFI to enter deep sleep, this product enters Deep sleep mode When set to 0, execute WFI to enter sleep, this product enters sleep/Idle mode	RW
1		When SLEEPONEXIT is set to 1, when exiting exception handling and returning to the program thread, the processor Automatically enter sleep mode (WFI) When set to 0, the feature will be automatically disabled	RW
0	RESERVED	Reserve	

After entering deep sleep, there are two options for the system clock after waking up. The clock entering deep sleep is used by default, and the configuration is controlled by the SYSCtrl0.wakeup_byRCH bit.

If SYSCtrl0.wakeup_byRCH is 1, no matter what clock is before entering deep sleep, it will be used after wakeup.

The internal high-speed clock RCH. If an external crystal oscillator is used, this setting can speed up the wake-up of the system.

Page 36

3 System Controller (SYSCTRL)

3.1 Clock source introduction

The clock control module mainly controls the system clock and peripheral clock, and can configure different clock sources as the system clock,

Different system clock dividers can be configured, and peripheral clocks can be enabled or disabled. In addition, in order to ensure the accuracy of the os

The internal clocks have calibration functions.

This product supports the following four different clock sources as the system clock:

Internal high-speed RC clock RCH (output frequency is 4~24MHz)

Internal low-speed RC clock RCL (38.4K and 32.768K configurable)

External high-speed crystal oscillator clock XTH

External low-speed crystal oscillator clock XTL

Notice:

– When switching the clock source of the system clock, please strictly follow the operation steps to switch, see section [3.2 for](#) details.

– XTL can directly input the 32.768KHz clock signal from the P14 pin without connecting the crystal oscillator. XTH can be unconnected

Crystal oscillator, directly input 4~32MHz clock signal from P01 pin.

This product also contains the following two auxiliary clocks:

Internal low-speed 10K clock; only for watchdog and CLKTRIM modules.

Internal 150K clock; only for LVD and VC modules.

The following figure shows the clock architecture of the product:

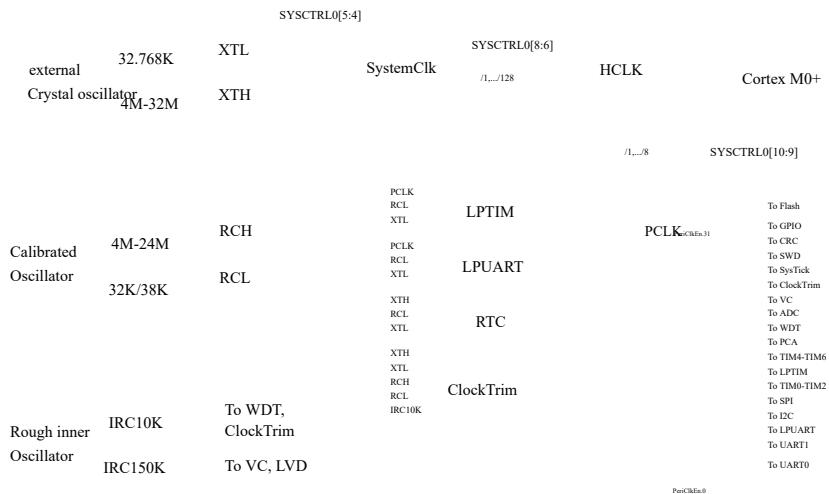


Figure 3-1 Block diagram of the clock control module

3.1.1 Internal high-speed RC clock RCH

The default clock source after the chip is powered on or reset is the internal high-speed clock with a frequency of 4MHz; when the system enters Deep Sleep, this high-speed clock will automatically turn off.

The output frequency of RCH can be adjusted by changing the value of register RCH_CR[10:0]. Each increase of register value by 1 The output frequency of RCH is increased by about 0.2%, and the total adjustment range is 4~24MHz.

The 5 frequencies 4MHz, 8MHz, 16MHz, 22.12MHz, 24MHz have been pre-tuned at the factory; if other frequencies are required Please adjust the value of this register manually.

To change the RCH output frequency, you need to follow a specific change sequence. For details, refer to the system clock switching chapter.

The internal high-speed clock only needs 4us from startup to stability. In order to respond quickly to interrupts in deep sleep mode, it is recommended to Switch the system clock to RCH before entering deep sleep mode.

3.1.2 Internal low-speed RC clock RCL

The output frequency of the internal low-speed clock can be selected through the register RCL_CR[9:0]. The available frequencies are 38.4KHz, 32.768KHz. When the system enters Deep Sleep, this low-speed clock will not automatically turn off, and the ultra-low power consumption Assume that the module can choose RCL as its clock.

3.1.3 External low-speed crystal oscillator clock XTL

The external low-speed crystal clock requires an external 32.768KHz low-power crystal oscillator, which has ultra-high precision and ultra-low power consumption.

When the system enters Deep Sleep, this low-speed clock will not automatically turn off. Peripheral modules working in ultra-low power mode can be

Choose XTL as its clock.

XTL can also directly input the 32.768KHz clock signal from the P14 pin without connecting the crystal oscillator. Input clock from P14

The signal method is: configure P14 pin as GPIO input; set SYSCTRL1. EXTL_EN to 1; set

SYSCTRL0. XTL_EN is 1.

Notice:

-The crystal and its matching devices must meet the relevant requirements of the low-speed external clock **XTL** in the electrical characteristics of the device.

3.1.4 External high-speed crystal oscillator clock **XTH**

The external high-speed crystal oscillator clock needs to be connected with a 4 MHz ~ 32MHz high-speed crystal oscillator. When the system enters Deep Sleep, this high-speed clock will be automatically turned off.

XTH can also directly input the 4MHz ~ 32MHz clock signal from the P01 pin without connecting the crystal oscillator. Input from P01

The method of clock signal is: configure P01 pin as GPIO input; set SYSCTRL1. EXTH_EN to 1; set

SYSCTRL0. XTH_EN to 1.

Notice:

-The crystal and its matching device must meet the relevant requirements of the high-speed external clock **XTH** in the electrical characteristics of the device.

3.1.5 Clock startup process

The above four clock sources all need start-up stabilization time. The following figure takes the external XTH as an example to illustrate the start-up stabilization process.

XTH_EN

XTH_O



Figure 3-2 Schematic diagram of crystal oscillator clock startup

3.2 System clock switching

The clock source of the system clock can be switched among RCH, RCL, XTH, XTL through SYSCTRL0[5:4].

The clock switching operation must be carried out in accordance with the standard clock switching process, otherwise an abnormality may occur. If the r

If the frequency is greater than 24MHz, you need to set FLASH_CR_WAIT to 1.

Note: To rewrite the value of FLASH_CR_WAIT, you need to write 0xA5A5, 0xA5A5 to the FLASH_BYPASS register first, and then

Then assign values to FLASH_CR_WAIT, see the chapter of FLASH controller for details.

3.2.1 Standard clock switching process

The operation process is as follows:

Step1: If the new clock source requires an external pin, set the pin to an appropriate mode.

Note: When connecting an external crystal oscillator, an analog pin is required; when connecting an external clock input, a GPIO input is required and the external clock input

Step2: Configure the oscillation parameters of the new clock source.

Step3: Enable the oscillator of the new clock source.

Step4: According to the higher frequency of the current clock source and the new clock source, follow the flow configuration of the Flash controller chapter FLASH_CR_WAIT.

Step5: Wait for the new clock source to output a stable frequency.

Step6: Configure SYSCTRL0. Clk_sw4_sel and select the source of the system clock as the new clock source.

Step7: According to the frequency of the new clock source, configure FLASH_CR_WAIT according to the procedure of the Flash controller chapter.

Step8: Turn off clock sources that are no longer in use.

3.2.2 Example of switching from RCH to XTL

The operation process is as follows:

Step1: Set P1ADS. P1ADS4 and P1ADS. P1ADS4 to 1, configure P14/P15 pins as analog ports.

Step2: According to the characteristics of the crystal oscillator, configure XTL_CR.Driver and XTL_CR.Startup.

Step3: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register sequentially to enable register rewriting.

Step4: Set SYSCTRL0. XTL_EN to 1, enable the crystal oscillator circuit.

Step5: The query waits for the XTL_CR. Stable flag to change to 1, and the crystal oscillator outputs a stable clock.

Step6: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register sequentially to enable register rewriting.

Page 41

Step7: Set SYSCTRL0. Clk_sw4_sel to 3, and switch the system clock to XTL.

Step8: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step9: Set SYSCTRL0. RCH_EN to 0, turn off the RCH oscillator.

3.2.3 Example of switching from RCH to XTH

The operation process is as follows:

Step1: Set P0ADS. P0ADS1 and P0ADS. P0ADS1 to 1, configure P01/P02 pins as analog ports.

Step2: According to the characteristics of the crystal oscillator, configure XTH_CR.Driver.

Step3: Set XTH_CR.Startup to 3, and select the longest crystal oscillator stabilization time.

Step4: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step5: Set SYSCTRL0. XTH_EN to 1, enable the crystal oscillator circuit.

Step6: According to the crystal oscillator frequency, configure FLASH_CR_WAIT.

Step7: After the query waits for the XTH_CR. Stable flag to change to 1, the software delays more than 10ms, and the crystal oscillator output is stable clock.

Step8: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step9: Set SYSCTRL0. Clk_sw4_sel to 1, and switch the system clock to XTH.

Step10: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step11: Set SYSCTRL0. RCH_EN to 0, turn off the RCH oscillator.

Page 42

The following figure shows the timing diagram of clock switching:

System clock: RCH switch XTH

External XTH start

External XTH

Internally sampled
XTH clock

XTH stable signal

Internal RCH enable
Signal

Internal RCH
clock

Clock switching signal
SYSCTRL0[5:4]

System clock

Figure 3-3 Schematic diagram of clock switching

3.2.4 Example of switching from **RCL** to **XTH**

The operation process is as follows:

Step1: Set P0ADS. P0ADS1 and P0ADS. P0ADS1 to 1, configure P01/P02 pins as analog ports.

Step2: According to the characteristics of the crystal oscillator, configure XTH_CR.Driver.

Step3: Set XTH_CR. Startup to 3, and select the longest crystal oscillator stabilization time.

Step4: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step5: Set SYSCTRL0. XTH_EN to 1, enable the crystal oscillator circuit.

Step6: According to the crystal oscillator frequency, configure FLASH_CR_WAIT.

- Step7: After the query waits for the XTH_CR. Stable flag to change to 1, the software delays more than 10ms, and the crystal oscillator output is stable clock.
- Step8: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.
- Step9: Set SYSCTRL0. Clk_sw4_sel to 1, and switch the system clock to XTH.

Page 43

- Step10: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.
- Step11: Set SYSCTRL0. RCL_EN to 0, turn off the RCL oscillator.

3.2.5 Example of switching from **RCH to **RCL****

The operation process is as follows:

- Step1: Configure RCL_CR.TRIM and RCL_CR. Startup.
- Step2: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register sequentially to enable register rewriting.
- Step3: Set SYSCTRL0. RCL_EN to 1 to enable the RCL oscillator circuit.
- Step4: The query waits for the RCL_CR. Stable flag to change to 1, and the RCL outputs a stable clock.
- Step5: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register sequentially to enable register rewriting.
- Step6: Set SYSCTRL0. Clk_sw4_sel to 2, and switch the system clock to RCL.
- Step7: If you need to turn off the RCH oscillator, perform subsequent operations: write 0x5A5A, 0xA5A5, enable register rewriting; set SYSCTRL0. RCH_EN to 0, turn off the RCH oscillator.

3.2.6 Example of switching from **RCL to **RCH****

The operation process is as follows:

- Step1: Configure RCH_CR.TRIM.
- Step2: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register sequentially to enable register rewriting.
- Step3: Set SYSCTRL0. RCH_EN to 1, enable RCH oscillator circuit.
- Step4: The query waits for the RCH_CR. Stable flag to change to 1, and RCH outputs a stable clock.
- Step5: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register sequentially to enable register rewriting.
- Step6: Set SYSCTRL0. Clk_sw4_sel to 0, and switch the system clock to RCH.
- Step7: If you need to turn off the RCL oscillator, perform subsequent operations: write 0x5A5A, 0xA5A5, enable register rewriting; set SYSCTRL0. RCL_EN to 0, turn off the RCL oscillator.

3.2.7 Switching between different oscillation frequencies of RCH

There are two schemes for switching between RCH different oscillation frequencies.

Option 1 :

Step1: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register sequentially to enable register rewriting.

Step2: Set SYSCTRL0.HCLK_PRS to 0x7.

Step3: Adjust the output frequency of RCH step by step up or down, 4M -> 8M -> 16M -> 24M/22.12M or 24M/22.12M -> 16M -> 8M -> 4M.

Step4: Write 0x5A5A and 0xA5A5 to the SYSCTRL2 register in sequence to enable register rewriting.

Step5: Set SYSCTRL0.HCLK_PRS to 0x0.

The sample code for switching from 4M to 24M is as follows:

```
M0P_SystemCtrl->SYSCTRL2 = 0X5A5A;  
  
M0P_SystemCtrl->SYSCTRL2 = 0XA5A5;  
  
M0P_SystemCtrl->SYSCTRL0_f.HCLK_PRS = 7;  
  
M0P_SystemCtrl->RCH_CR = *((uint16 *) (0X00100C08)); //4M  
  
M0P_SystemCtrl->RCH_CR = *((uint16 *) (0X00100C06)); //8M  
  
M0P_SystemCtrl->RCH_CR = *((uint16 *) (0X00100C04)); //16M  
  
M0P_SystemCtrl->RCH_CR = *((uint16 *) (0X00100C00)); //24M  
  
M0P_SystemCtrl->SYSCTRL2 = 0X5A5A;  
  
M0P_SystemCtrl->SYSCTRL2 = 0XA5A5;  
  
M0P_SystemCtrl->SYSCTRL0_f.HCLK_PRS = 0;
```

Option 2 :

Step1: Switch the system clock to RCL, see 3. [2.5](#) Example of switching from RCH to RCL.

Step2: Change the value of RCH_CR.TRIM and change the frequency of RCH oscillation.

Step3: The system clock is switched to RCH, see 3. [2.6](#) switched from the low speed to the inside RCL exemplary RCH.

3.3 Clock calibration module

This product has a built-in clock calibration circuit. As shown in the figure below, the four sources of the system clock can be calibrated with each other.

After the reference clock and the calibrated clock, set the register REFCNT value, set cali.start to start the clock calibration circuit

At this time, two 32-bit counters (increment and decrement) work at the same time. When the decrement counter is equal to 0, cali.finish

It is set to indicate the end of calibration. At this time, the software can read the value of CALCNT, so that it is easy to get the reference clock

The frequency relationship with the calibrated clock.

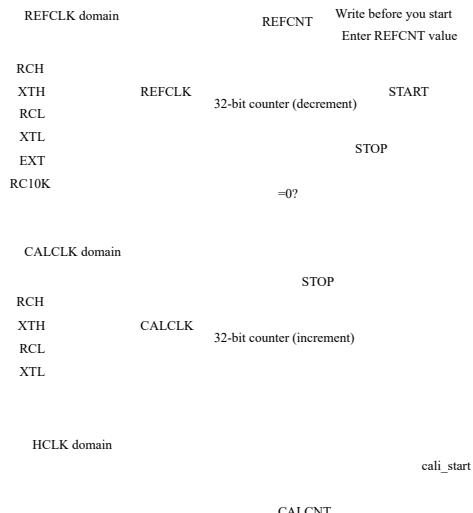


Figure 3-4 Schematic diagram of clock calibration

3.4 Interrupt wake-up control

When the processor executes the WFI instruction and enters the sleep state, it will stop executing instructions. An interrupt occurred during sleep. When a request (higher priority) is required and needs to be processed, the processor will be awakened.

The behavior of the processor in the sleep state when receiving an interrupt request is shown in the following table:

PRIMASK status	WFI behavior	wake	ISR execution
0	IRQ priority > current level	Y	Y

0	IRQ priority \leq current level	N	N
1	IRQ priority $>$ current level	Y	N
1	IRQ priority \leq current level	N	N

3.4.1 Method of executing interrupt service routine after waking up from deep sleep mode

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set SCB->SCR.SLEEPDEEP to 1
4. Execute WFI command to enter deep sleep mode
5. The system enters deep sleep mode and waits for the interrupt to wake up, and executes the interrupt service routine after wake up

Routine:

```
SCB_SCR |= 0x00000004u;
while(1)
{
    __asm("WFI");
}
```

3.4.2 The method of not executing the interrupt service routine after waking up from the deep sleep mode

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set PRIMASK to 1
4. Set SCB->SCR.SLEEPDEEP to 1

5. Execute WFI command to enter Deep Sleep mode
6. The system enters deep sleep mode and waits for the interrupt to wake up, and executes the next instruction after wake up
7. Clear the interrupt flag, clear the interrupt pending state

Routine:

```
__asm("CPSID I"); //Set PRIMASK
SCB_SCR |= 0x00000004u;
while(1)
{
    __asm("WFI");
    BTIMERLP_REG->TFCR_f.TFC=0;           //Clear Int Flag
    NVIC_ClearPendingIRQ(BASE_TIMER3_IRQn); //Clear Pending Flag
}
```

3.4.3 Use exit hibernation feature

Quitting sleep-on-exit is very suitable for interrupt-driven applications. When this feature is enabled, as long as the exception is completed

Normal processing and returning to thread mode, the processor will enter sleep mode. Using the exit hibernation feature, the processor

Can be in sleep mode as much as possible.

Cortex-M0 uses the exit sleep feature to enter sleep. This situation is the same as executing WFI immediately after exiting abnormally.

The effect is similar. However, in order to save the stack operation when entering the exception next time, the processor will not execute the output

The process of stacking.

1. Enable the NVIC corresponding to the module that needs to wake up the processor
2. Enable the interrupt corresponding to the module that needs to wake up the processor
3. Set SCB->SCR.SLEEPDEEP to 1
4. Set SCB—>SCR.SLEEPONEXIT to 1
5. Execute WFI command to enter Deep Sleep mode
6. The system enters deep sleep mode and waits for the interrupt to wake up, and executes the interrupt service subroutine after wake up
7. Automatically enter sleep mode when exiting interrupt service

Routine:

```
SCB_SCR |= 0x00000004u;  
SCB_SCR |= 0x00000002u;  
while(1)  
{  
    __asm("WFI");  
}  
}
```

Page 49

3.5 Register

Base address 0x40002000

register	Offset address	describe
SYSCTRL0	0x000	System control register 0
SYSCTRL1	0x004	System Control Register 1
SYSCTRL2	0x008	System Control Register 2
RCH_CR	0x00C	RCH control register
XTH_CR	0x010	XTH control register
RCL_CR	0x014	RCL control register
XTL_CR	0x018	XTL control register
PERI_CLKEN	0x020	Peripheral module clock control register
SYSTICK_CR	0x034	Systick clock control

Table 3-1 System control register table

Page 50**3.5.1 System Control Register 0 (SYSCTRL0)**

Offset address: 0x000

Reset value: 0x0000 0001

31	30	29	28	27	26	25	twenty fourty twenty twenty twenty off	19	18	17	16
Reserved											
15	14	13	12	11	10	9	8	7	6	5	4
Wakeup_						PCLK_PRS	HCLK_PRS		clk_sw4_sel	XTL_	RCL_
byRCH										EN	EN
R/W						R/W	R/W		R/W		R/W

Bit	mark	Function description
31:16	Reserved	Reserve
15	wakeup_	1: After waking up from Deep Sleep, the system clock source is RCH, and the original clock continues to be enabled. 0: After waking up from Deep Sleep, the system clock source is not changed.
14:11	byRCH	
14:11	Reserved	Reserve
		PCLK frequency division selection
		00: HCLK
10:9	PCLK_PRS	01: HCLK/2 10: HCLK/4 11: HCLK/8
		HCLK frequency division selection
		000: SystemClk 001: SystemClk/2 010: SystemClk/4
8:6	HCLK_PRS	011: SystemClk/8 100: SystemClk/16 101: SystemClk/32 110: SystemClk/64 111: SystemClk/128
		System clock source selection
		00: Internal high-speed clock RCH
5:4	Clk_sw4_sel	01: External high-speed crystal oscillator XTH 10: Internal low-speed clock RCL 11: External low-speed crystal oscillator XTL
		External low-speed crystal oscillator XTL enable control
3	XTL_EN	0: off 1: Enable Note: P14 and P15 need to be set as analog ports.

Page 51

		Internal low-speed clock RCL enable control
2	RCL_EN	0: off 1: Enable
		External high-speed crystal oscillator XTH enable control
1	XTH_EN	0: off 1: Enable
		Note: When the system enters DeepSleep, the high-speed clock will be automatically turned off.
		The internal high-speed clock RCH enable signal.
0	RCH_EN	0: off 1: Enable
		Note: When the system enters DeepSleep, the high-speed clock will be automatically turned off.

Notice:

- Every time you rewrite the values of SYSCTRL0 and SYSCTRL1, you need to write 0x5A5, 0x5A5, SYSCTRL2, and 0xA5A5. Such steps can effectively prevent the misoperation of the SYSCTRL0 and SYSCTRL1 registers.

Page 52

3.5.2 System Control Register 1 (SYSCTRL1)

Offset address: 0x004

Reset value: 0x00000008

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16		
Reserved																
15	14	13		12	11	10	9	8	7	6	5	4	3	2	1	0
								SWD_	RES_	LOCK	RTC_		XTL_alwa	EXTL	EXTH	
								UIO	UIO	_EN	LPW	Res	yson	_EN	_EN	Re
								R/W	R/W	R/W	R/W		R/W	R/W	R/W	s

Bit	mark	Function description
31:12	Reserved	
11:9	RTC_FREQ_ADJUST	RTC high-speed clock compensation clock frequency selection 000 4M; 001 6M; 010 8M; 011 12M 100 16M; 101 20M; 110 24M; 111 32M;
8	SWD_USE_IO	SWD port function configuration 0: SWD port 1: GPIO port
7	RESET_USE_IO	RESET port function configuration 0: RESET port 1: GPIO port
6	LOCKUP_EN	Cortex-M0+ LockUp function configuration 0: off 1: enable Note: After enabling, the CPU will reset the MCU when it reads an invalid instruction, which can enhance the system reliability.
5	RTC_LPW	RTC module low power control 1: Low power consumption mode is enabled 0: Low power consumption mode disabled Note: After being enabled, the RTC module enters a low-power state, and its registers cannot be read or written.
4	Reserved	
3	XTL_ALWAYS_ON	XTL advanced enable control 1: SYSCTRL0.XTL_EN can only be set. 0: SYSCTRL0.XTL_EN can be set and cleared. External XTL clock input control
2	EXTL_EN	1: XTL output clock is input from P14. 0: XTL output clock is generated by crystal oscillator. Note: When using P14 to input the clock, you need to set SYSCTRL0.XTL_EN to 1.
0	Reserved	

1	EXTH_EN	External XTH input control 1: XTH output clock is input from P01. 0: XTH output clock is generated by crystal oscillator. Note: When using P01 to input the clock, you need to set SYSCTRL0.XTH_EN to 1.
0	Reserved	

Notice:

- Every time you rewrite the values of SYSCTRL0 and SYSCTRL1, you need to write 0x5A5, 0x5A5, SYSCTRL2, and 0xA5A5. Such steps can effectively prevent the misoperation of the SYSCTRL0 and SYSCTRL1 registers.

Page 54**3.5.3 System Control Register 2 (SYSCTRL2)**

Offset address: 0x008

Reset value: 0x00000000

31	30	29	28	27	26	25	twenty	four	twenty	three	twenty	two	ten	19	18	17	16
Reserved																	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
SYSCTRL2																	

Bit	mark	Function description
31:16	Reserved	

Register SYSCTL0, SYSCTRL1 protect series control register, write SYSCTRL2 first

0xA5A5, then write 0xA5A5 to start the write operation to the registers SYSCTL0 and SYSCTRL1, only

15:0 SYSCTRL2

To write to the registers SYSCTRL0 and SYSCTRL1, this protection bit will automatically restore the protection status.

Need to rewrite the series to open the protection.

Page 55

3.5.4 RCH control register (RCH_CR)

Offset address: 0x00C

Reset value: 0x00000126

31	30	29	28	27	26	25	twenty	fourty	thirty	twenty	two	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				Stable				TRIM							

Bit	mark	Function description
-----	------	----------------------

31:12 Reserved

The internal high-speed clock RCH stability flag.

11 stable

1: Represents that RCH has been stabilized and can be used by internal circuits.

0: Represents that RCH is not stable and cannot be used by internal circuits.

Clock frequency adjustment, the output frequency of RCH can be adjusted by changing the value of this register. Register value

Each increase of 1 will increase the output frequency of RCH by about 0.2%, and the total adjustment range is 4~24MHz.

5 sets of frequency calibration values have been saved in Flash, read and write the calibration values in Flash

RCH_CR.TRIM can get accurate frequency.

10:0 TRIM

24M calibration value address: 0x00100C00-0x00100C01

22.12M Calibration value address: 0x00100C02-0x00100C03

16M calibration value address: 0x00100C04-0x00100C05

8M calibration value address: 0x00100C06-0x00100C07

4M calibration value address: 0x00100C08-0x00100C09

RCH required output frequency changes in accordance with a specific timing, see the system clock switch 3.2 [Section 7](#).**Page 56****3.5.5 Oscillation XTH Control Register (XTH_CR)**

Offset address: 0x010

Reset value: 0x00000022

31	30	29	28	27	26	25	twent	twent	twent	twent	on	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
									Stable	Startup		Driver			
									RO	R/W		R/W			

Bit mark Function description

31:6 Reserved

External high-speed clock XTH stability flag.

1: Represents XTH has been stabilized and can be used by internal circuits.

6 stable

0: XTH is not stable and cannot be used by internal circuits.

Note: In order to increase the reliability of the system, the software needs to delay more than 10ms after the logo is inquired.

Only then can the system clock be switched to XTH.

External high-speed clock XTH stabilization time selection

00: 256 cycles;

01: 1024 cycles;

5:4 Startup

10: 4096 cycles;

11: 16384 cycles;

Note: It is strongly recommended to set the XTH stabilization time to 11. If the XTH stabilization time is insufficient,

The system cannot work stably when switching clocks or waking up from deep sleep.

3:2 Freq selects the operating frequency of the crystal oscillator

11: 24M~32M

10: 16M~24M

01: 8M~16M

00: 4M~8M

3:0 Driver

1:0 Driver Select the driving capability of the crystal oscillator

11: The strongest driving ability

10: Default drive capability (recommended value)

01: Weak driving ability

00: Weakest driving ability

Note: It is necessary to select the appropriate drive energy according to the characteristics of the crystal oscillator, the load capacitance and the parasitic parameter. The greater the driving capability, the greater the power consumption; the weaker the driving capability, the lower the power consumption.

Page 57

3.5.6 RCL control register (RCL_CR)

Offset address: 0x014

Reset value: 0x00000033Fh

31	30	29	28	27	26	25	24	twenty	thirty	twenty	twenty	on off	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reserved				Stable		Startup		TRIM								
				RO		R/W		R/W								

Bit	mark	Function description
-----	------	----------------------

31:13 Reserved

Internal low-speed clock RCL stable flag.

12 stable

1: It means that the RCL is stable and can be used by the internal circuit.

0: Represents that RCL is not stable and cannot be used by internal circuits.

Internal low-speed clock RCL stabilization time selection

11: 256 cycles;

11:10 Startup

10: 64 cycles;

01: 16 cycles;

00: 4 cycles;

Internal low-speed clock frequency adjustment, 2 sets of frequency calibration values are saved in Flash.

Read the calibration value in Flash and write it into RCL_CR.TRIM to get the precise frequency.

9:0 TRIM

38.4K calibration value address: 0x00100C20-0x00100C21

32.768K calibration value address: 0x00100C22-0x00100C23

Page 58**3.5.7 XTL control register (XTL_CR)**

Offset address: 0x018

Reset value: 0x00000021

31	30	29	28	27	26	25	twenty	fourty	thwenty	twenty	twenty	on@	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reserved																
									Stable	Startup		Driver				
									RO	R/W		R/W				

Bit mark Function description

31:6 Reserved

External low-speed crystal oscillator XTL stable flag.

6 stable 1: Represents XTL is stable and can be used by internal circuits.
0: XTL is not stable and cannot be used by internal circuits.

External low-speed crystal XTL stabilization time selection

00: 256 cycles;

5:4 Startup 01: 1024 cycles;
10: 4096 cycles;
11: 16384 cycles;
External low-speed crystal oscillator XTL drive selection
1111: Maximum drive
0000: minimum drive
3:2 Amp_control The fine adjustment of XTL oscillation amplitude.
11: Maximum amplitude
10: Larger amplitude (recommended value)
01: normal amplitude
00: minimum amplitude3:0 Driver 1:0 Driver External crystal oscillator drive capability selection
11: The strongest driving ability

10: Strong driving ability

01: Default drive capability (recommended value)

00: Weakest driving ability,

Note: It is necessary to select the appropriate drive energy according to the characteristics of the crystal oscillator, the load capacitance and the parasitic parameter. The greater the driving capability, the greater the power consumption; the weaker the driving capability, the lower the power consumption. 00: Weakest driving force

Page 59

3.5.8 Peripheral module clock control register (PERI_CLKEN)

Reset value: 0xC080_0000

Offset address: 0x020

31	30	29	28	27	26	25	twenty four	twenty five	twenty six	twenty seven	twenty eight	19	18	17	16
Flash			GPIO	Res.	CRC	Res.	TICK		Res.	Trim RTC		Res.	VC ADC		
R/W			R/W		R/W		R/W			R/W	R/W		R/W	R/W	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDT PCA					ADV	LP	BASE			SPI	Res. I2C		LPUART	UART1	UART0
R/W					Res.	TIM	TIM	TIM	Res.	R/W	R/W	R/W	R/W	R/W	R/W
						R/W	R/W	R/W		R/W	R/W		R/W	R/W	R/W

Bit	mark	Function description
31	flash	Flash controller module clock enable, you need to enable the clock when operating the flash register 1: enable; 0: disable Note: This bit is not affected when executing the program from the flash.
30:29	Res.	Reserved bit
28	GPIO	GPIO module clock enable. 1: enable; 0: disable
27	Res.	Reserved bit
26	CRC	CRC module clock enable. 1: enable; 0: disable
25	Res.	Reserved bit
24	TICK	SysTick timer reference clock enable. 1: enable; 0: disable
23:22	Res.	Reserved bit
21	Trim	CLKTRIM module clock enable. 1: enable; 0: disable
20	RTC	The RTC module clock is enabled. 1: enable; 0: disable
19:18	Res.	Reserved bit
17	VC	VC, LVD, module clock enable. 1: enable; 0: disable
16	ADC	ADC module clock enable. 1: enable; 0: disable
15	WDT	WDT module clock enable. 1: enable; 0: disable

14	PCA	PCA module clock enable. 1: enable; 0: disable
13:11	Res.	Reserved bit
10	ADVTIM	Timer456 module clock enable. 1: enable; 0: disable
9	LPTIM	LPTimer module clock enable. 1: enable; 0: disable
8	BASETIM	Timer012 module clock enable. 1: enable; 0: disable

7	Res.	Reserved bit
6	SPI	SPI module clock enable. 1: enable; 0: disable
5	Res.	Reserved bit
4	I2C	I2C module clock enable. 1: enable; 0: disable
3	Res.	Reserved bit
2	LPUART	LPUART module clock enable. 1: enable; 0: disable
1	UART1	UART1 module clock enable. 1: enable; 0: disable
0	UART0	UART0 module clock enable. 1: enable; 0: disable

HC32L110 Series User Manual Rev2.31

Page 60 of 527

Page 61

3.5.9 Systick clock control (SYSTICK_CR)

Reset value: 0x0100_0147

Offset address: 0x034

Bit	mark	Function description
31-28	Reserved	

		SysTick external reference clock selection 00: External low-speed clock XTL
27-26	CLK_SEL	01: Internal low-speed clock RCL 10: System clock divided by 8 SystemClk/8 11: External high-speed clock XTH
		SysTick clock source selection 1: Internal high frequency clock HCLK 0: External reference clock, selected by SYSTICK_CR[27:26]
25	NOREF	Note: When using an external reference clock, the reference clock frequency is not allowed to be higher than HCLK
		STCALIB accuracy indicator
twenty four	SKEW	1: STCALIB value represents roughly 10ms 0: STCALIB value represents accurate 10ms
23:0	STCALIB	10 millisecond calibration value when the reference clock is XTL

Page 62

4 Reset controller (**RESET**)

4.1 Introduction to Reset Controller

This product has 7 reset signal sources, each reset signal can make the CPU run again, most of the registered

The device will be reset to the reset value, and the program counter PC will be reset to point to 00000000.

POR/BOR reset (VCC domain and Vcore domain)

External Reset PAD reset

WDT reset

PCA reset

LVD reset

Cortex-M0+ SYSRESETREQ software reset

Cortex-M0+ LOCKUP hardware reset

Each reset source is indicated by the corresponding reset flag. The reset flags are all set by hardware and need to be cleared by user software.

When the chip is reset, if Reset_flag. POR15V or Reset_flag. POR5V is 1, it is a power-on reset.

The user program should clear the register Reset_flag at power-on reset, then reset_flag can be used during the next reset

The relevant bits of to determine the source of the reset.

The following figure describes the reset source of each area.

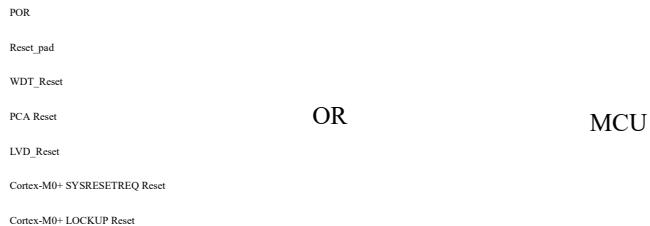


Figure 4-1 Schematic diagram of reset source

4.1.1 Power-on and power-off reset POR/BOR

This product has two power supply areas: **VCC** area and **Vcore** area. All analog modules and **IO** work on **VCC**

Area; other modules work in the **Vcore** area.

When the VCC area is powered on, when the VCC voltage is lower than the POR threshold voltage (typical value is 1.65V), POR5V will be generated Signal; when the VCC area is powered off, when the VCC voltage is lower than the BOR threshold voltage (typical value is 1.5V), it will generate POR5V signal.

When the Vcore area is powered on, when the Vcore voltage is lower than the POR threshold voltage, a POR15V signal will be generated; Vcore

When the area is powered off, when the Vcore voltage is lower than the BOR threshold voltage, a POR15V signal will be generated.

Both the POR5V signal and the POR15V signal will reset the chip's registers to the initialization state.

4.1.2 External reset pin reset

When the external reset pin detects a low level, a system reset is generated. The reset pin has built-in pull-up resistor, and

A glitch filtering circuit is integrated. The glitch filter circuit will filter glitch signals less than 20us (typical value), because

Therefore, the low-level signal added to the reset pin must be greater than 20us to ensure reliable chip reset.

4.1.3 WDT reset

Watchdog reset, please refer to the description of WDT chapter.

4.1.4 PCA reset

PCA reset, please refer to the description of PCA chapter.

4.1.5 LVD low voltage reset

For LVD reset, please refer to the description of LVD chapter.

4.1.6 Cortex-M0+ SYSRESETREQ reset

Cortex-M0+ software reset

4.1.7 Cortex-M0+ LOCKUP reset

When Cortex-M0+ encounters a serious exception, it will stop its PC pointer at the current address and lock itself

It resets the entire CORE area after a delay of several clock cycles.

Page 65

4.2 Register

4.2.1 Reset Flag Register (**Reset_flag**)

Reset value: 00000000_00000000_00000000_xxxxxx11b

Address: 0x4000201C

31	30	29	28	27	26	25	twenty	fourty	thirty	twenty	twenty	on off	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reserved																
							RSTB	sysreq	lockup	PCA	WDT	LVD	Por15	Por5v		
							RW0	RW0	RW0	RW0	RW0	RW0	RW0	RW0	RW0	

Bit	mark	Function description
31:8	Reserved	
7	RSTB	RESETB port reset flag, need to be initialized and cleared by software, power-on state is uncertain 1: Port reset occurred 0: No port reset occurred
6	Sysreq	Cortex-M0+ CPU software reset flag, need to be initialized and cleared by software, the power-on state is uncertain 1: Cortex-M0+ CPU software reset occurred 0: No Cortex-M0+ CPU software reset occurs
5	Lockup	Cortex-M0+ CPU Lockup reset flag, need to be initialized and cleared by software, power-on state is uncertain 1: Cortex-M0+ CPU Lockup reset occurred 0: No Cortex-M0+ CPU Lockup reset occurs
4	PCA	PCA reset flag, need to be initialized and cleared by software, power-on state is uncertain 1: PCA reset occurred 0: No PCA reset occurred
3	WDT	WDT reset flag, need to be initialized and cleared by software, power-on state is uncertain 1: WDT reset occurred 0: No WDT reset occurs
2	LVD	LVD reset flag, need to be initialized and cleared by software, power-on state is uncertain 1: LVD reset occurred 0: No LVD reset occurred
1	POR15V	Vcore domain reset flag 1: The Vcore domain is reset 0: No reset occurs in the Vcore domain
0	POR5V	VCC power domain reset flag 1: The VCC power domain is reset 0: No reset occurs in the VCC power domain

Page 66

4.2.2 Peripheral module reset control register (PERI_RESET)

Reset value: 0xD7B3C757

Address: 0x40002028

31	30 29 28		27	26	25	24	23	22	21	20	19	18	17	16
		GPIO	CRC	TICK		TRIM	RTC					VC ADC		
Res.		R/W	Res.	Res.	R/W	Res.	R/W	R/W		Res.		R/W	R/W	
15	14 13 12		11	10	9	8	7	6	5	4	3	2	1	0
			ADV	LP	BASE							LPUA		
Res.	PCA.	Res.	TIM	TIM	TIM	SPI		I2C		Res.	RT		UART1	UART0
			R/W	R/W	R/W							R/W	R/W	R/W

Bit	mark	Function description
31:29	Res.	Reserved bit
28	GPIO	GPIO module reset enable. 1: Normal operation; 0: Module is in reset state
27	Res.	Reserved bit
26	CRC	CRC module reset enable. 1: Normal operation; 0: Module is in reset state
25	Res.	Reserved bit
twenty four	TICK	SYSTICK module reset enable.
		1: Normal operation; 0: Module is in reset state
23:22	Res.	Reserved bit
twenty one	TRIM	CLKTRIM module reset enable.
		1: Normal operation; 0: Module is in reset state
20	RTC	RTC module reset enable. 1: Normal operation; 0: Module is in reset state
19-18	Res.	Reserved bit
17	VC	VC, LVD, module reset enable. 1: Normal operation; 0: Module is in reset state
16	ADC	ADC module reset enable. 1: Normal operation; 0: Module is in reset state
15	Res.	Reserved bit
14	PCA	PCA module reset enable.
		1: Normal operation; 0: Module is in reset state
13:11	Res.	Reserved bit
10	ADVTIM	Timer456 module reset enable.
		1: Normal operation; 0: Module is in reset state

9	LPTIM	LPTimer module reset enable. 1: Normal operation; 0: Module is in reset state
8	BASETIM	Timer012 module reset enable. 1: Normal operation; 0: Module is in reset state
7	Res.	Reserved bit
6	SPI	SPI module reset enable. 1: Normal operation; 0: Module is in reset state
5	Res.	Reserved bit

4	I2C	I2C module reset enable. 1: Normal operation; 0: Module is in reset state
3	Res.	Reserved bit
2	LPUART	LPUART module reset enable. 1: Normal operation; 0: Module is in reset state
1	UART1	UART1 module reset enable. 1: Normal operation; 0: Module is in reset state
0	UART0	UART0 module reset enable. 1: Normal operation; 0: Module is in reset state

5 Interrupt Controller (NVIC)

5.1 Overview

The Cortex-M0+ processor has a built-in nested vectored interrupt controller (NVIC), which supports up to 32 interrupt requests (IRQ)

Input, and 1 non-maskable interrupt (NMI) input (not used in this product system). In addition, deal with

The processor also supports multiple internal exceptions.

Each exception source has a separate exception number, and each exception type has a corresponding priority. Some exceptions have advantages.

The advanced stage is fixed, while some are programmable. The details are shown in the following table:

Exception num	Exception type	priority	describe
1	Reset	-3 (highest)	Reset
2	NMI	-2	Non-maskable interrupt (not used in this system)
3	Hardware error	-1	Error handling exception
	Reserve		...

4-10 11	SVC	NA Programmable	Call the management program through SVC instruction
12-13	Reserve	NA	...
14	PendSV	Programmable	Suspendable requests for system services
15	SysTick	Programmable	SysTick timer
16	Interrupt #0	Programmable	External interrupt #0
17	Interrupt #1	Programmable	External interrupt #1
...
47	Interrupt #31	Programmable	External interrupt #31

Table 5-1 Cortex-M0+ processor exception list

This chapter only gives a detailed introduction to the 32 external interrupt requests (interrupt #0 to interrupt #31) of the processor.

For the specific circumstances of the exception, please refer to other related documents. At the same time, this chapter only discusses the interrupt of the N

The processing mechanism, the interrupt generation mechanism of the peripheral module itself is not discussed here.

5.2 Interrupt priority

Each external interrupt corresponds to a priority register, each priority is 2 bits wide, and uses interrupt priority

The highest two bits of the level register, each register occupies 1 byte (8 bits). Under this setting, the ones that can be used take precedence

The levels are 0x00 (the highest), 0x40, 0x80, and 0xc0 (the lowest).

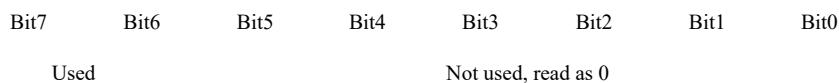


Figure 5-1 Only the upper two bits of the priority register are used

Page 69

If the processor is already running another interrupt processing, and the priority of the new interrupt is higher than the one being executed, then Preemption will occur. The interrupt processing that is running will be suspended and a new interrupt will be executed. This process is usually referred to as Interrupt nesting. After the new interrupt is executed, the previous interrupt processing will continue to execute and return to In the program thread.

If the priority of another interrupt processing the processor is running is the same or higher, the new interrupt will wait and And enter the suspended state. The pending interrupt will wait until the current interrupt level is changed, for example, the currently running interrupt After the processing completes and returns, the current priority is reduced to a level lower than the pending interrupt.

If two interrupts occur at the same time and their priority is the same, the interrupt with the smaller interrupt number will be executed first.

For example, if interrupt #0 and interrupt #1 are enabled and have the same priority, when they are triggered at the same time, interrupt #0 will be Execute first.

5.3 Interrupt vector table

When the Cortex-M0+ processor needs to process an interrupt service request, it needs to first determine the start address of the exception handling, so

The required information is called a vector table, as shown in Figure 5-2 . The vector table is stored at the beginning of the memory space and contains the

The exception (interrupt) vector of the exception (interrupt) available in the system, and the initial value of the main stack pointer (MSP).

Memory address Exception number

0x00000048	Interrupt #2 vector	18
0x00000044	Interrupt #1 vector	17
0x00000040	Interrupt #0 vector	16
0x0000003C	SysTick vector	15
0x00000038	PendSV vector	14
0x00000034	Unused	13
0x00000030	Unused	12
0x0000002C	SVC vector	11
0x00000028	Unused	10
0x00000024	Unused	9
0x00000020	Unused	8
0x0000001C	Unused	7
0x00000018	Unused	6
0x00000014	Unused	5
0x00000010	Unused	4
0x0000000C	Hardware error exception	3
0x00000008	NMI vector	2
0x00000004	Reset vector	1
0x00000000	MSP initial value	0

Figure 5-2 Interrupt vector table

Among them, the storage order of the interrupt vector is the same as the interrupt number. Since each vector is 1 word (4 bytes), the interrupt address of the vector is the interrupt number multiplied by 4. Each interrupt vector is the starting address of the interrupt processing.

5.4 Interrupt input and suspend behavior

In the NVIC module of the Cortex-M0+ processor, each interrupt input corresponds to a suspend status register, and each register has only 1 bit, which is used to save the interrupt request, regardless of whether the request is confirmed or not. When the processor starts to process this interrupt, the hardware will automatically clear the pending status bit. The peripherals of this system use level-triggered interrupt output. When an interrupt event occurs, because the peripheral is connected to the NVIC, the interrupt signal will be confirmed. Before the processor executes the interrupt service and clears the interrupt signal of the peripheral, the signal will keep it high. Inside the NVIC, when an interrupt is detected, the pending state of the interrupt will be set. After the processor receives the interrupt and starts to execute the interrupt service routine, the suspended state will be cleared. The process is shown in Figure 5-3.

Shown:

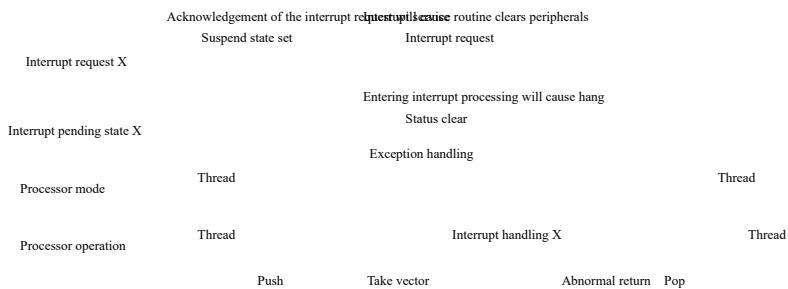


Figure 5-3 Interrupt activation and suspension status

If the interrupt request is not executed immediately and is cleared by the software before the confirmation, the processor will ignore this request, and interrupt processing will not be performed. The interrupt pending status can be cleared by writing to the NVIC_CLRPEND register. This kind of processing is very useful when setting up a peripheral, because before the setting, the peripheral may have generated a medium request.

If the peripheral is still holding the interrupt request when the software clears the suspend state, the suspend state will be generated immediately. It's time

The process is shown in Figure 5-4 :

Page 71

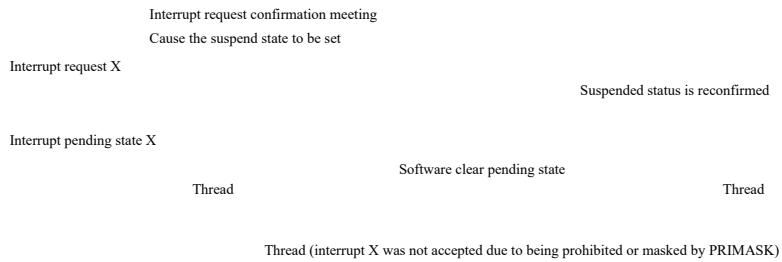


Figure 5-4 The interrupt pending state is cleared and then reconfirmed

If the interrupt request generated by the peripheral is not cleared during exception handling, the suspended state will be re-suspended after the exception re-

Activate, so that the interrupt service routine will be executed again. The process is shown in Figure 5-5 :

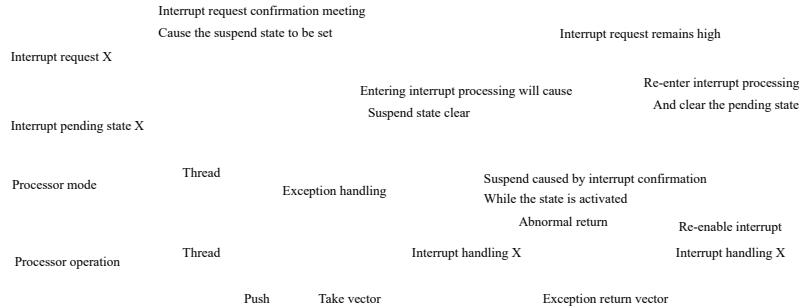


Figure 5-5 If the interrupt request remains high when the interrupt exits, it will cause the interrupt processing to be executed again

Page 72

If a peripheral interrupt request is generated during the execution of the terminal service program, the request will be treated as a new interrupt request.

And after this interruption exits, it will cause the interrupt service routine to be executed again. The process is shown in Figure 5-6 :

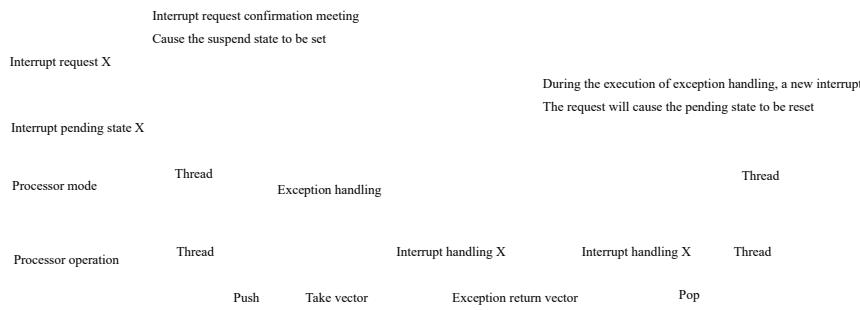


Figure 5-6 Interrupt pending during interrupt processing can also be confirmed

Page 73

5.5 Interrupt waiting

Normally, the interrupt waiting time of NVIC is 16 cycles. This wait time is from the interrupt acknowledged processor time

The clock cycle starts and continues until the interrupt processing starts and the execution ends. The following prerequisites are required to calculate interrupt waiting time:

The interrupt is enabled and is not shielded by SCS_PRIMASK or other exception handling being executed.

The memory system does not have any waiting state, and the fetching at the beginning of interrupt processing, stack pushing, vector fetching or interrupt return address calculation does not cause any waiting state.

Refers to the bus transfer, if the memory system needs to wait, then the wait state generated when the bus transfer occurs will be added to the interrupt waiting time.

The state may delay the interrupt.

The following situations may cause different interrupt waiting:

The end of the interrupt is chained. If another interrupt request is generated when the interrupt returns, the processor will skip popping and pushing the stack, thus reducing the interrupt waiting time.

The process of pushing the stack, thus reducing the interrupt waiting time.

Delayed arrival, if an interrupt occurs, another low-priority interrupt is being pushed onto the stack, due to the delay in the interrupt response.

In the presence of the arrival mechanism, the high-priority interrupt will be executed first, which will also cause the waiting time of the high-priority interrupt to be reduced.

5.6 Interrupt source

Because the NVIC of the Cortex-M0+ processor supports up to 32 external interrupts, and in this system, the external interrupt source

More than 32, so some external interrupts are multiplexed on the same NVIC interrupt input, and NMI (non-maskable

Interrupt) is not used. The corresponding relationship between all external interrupt sources of this system and NVIC interrupt input is shown in the following table:

NVIC interrupt input	External interrupt source	Active mode	Sleep mode	DeepSleep mode
Interrupt #0	PORT0	v	v	v
Interrupt #1	PORT1	v	v	v
Interrupt #2	PORT2	v	v	v
Interrupt #3	PORT3	v	v	v
Interrupt #4	Reserved	-	-	-
Interrupt #5	Reserved	-	-	-
Interrupt #6	UART0	v	v	-
Interrupt #7	UART1	v	v	-
Interrupt #8	LPUART	v	v	v
Interrupt #9	Reserved	-	-	-
Interrupt #10	SPI	v	v	-
Interrupt #11	Reserved	-	-	-

Interrupt #12	I2C	v	v	-
Interrupt #13	Reserved	-	-	-
Interrupt #14	TIM0	v	v	-
Interrupt #15	TIM1	v	v	-
Interrupt #16	TIM2	v	v	-
Interrupt #17	LPTIM	v	v	v
Interrupt #18	TIM4	v	v	-

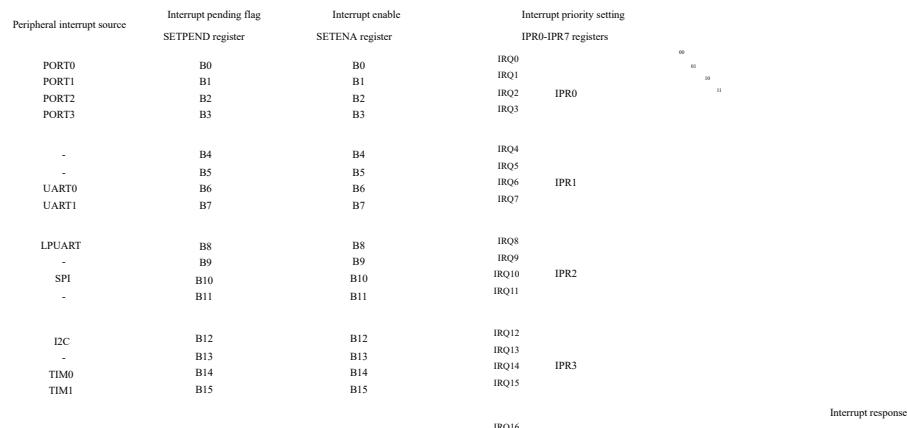
Interrupt #19	TIM5	v	v	-
Interrupt #20	TIM6	v	v	-
Interrupt #21	PCA	v	v	-
Interrupt #22	WDT	v	v	v
Interrupt #23	RTC	v	v	v
Interrupt #24	ADC	v	v	-
Interrupt #25	Reserved	-	-	-
Interrupt #26	VC0	v	v	v
Interrupt #27	VC1	v	v	v
Interrupt #28	LVD	v	v	v
Interrupt #29	Reserved	-	-	-
Interrupt #30	EFCTRL/RAMCTRL v		v	-
Interrupt #31	CLK_TRIM	v	v	v

Table 5-2 Correspondence between external interrupt and NVIC interrupt input

Notice:

- Because some module interrupts are multiplexed to the same IRQ interrupt source, when the CPU enters the interrupt operation, it must

First determine which module generated the interrupt, and then perform the corresponding interrupt operation.

5.7 Interrupt structure diagram

TIM2 LPTIM	B16 B17	B16 B17	IRQ17	
TIM4	B18	B18	IRQ18	
TIM5	B19	B19	IRQ19	
				IPR4
TIM6	B20	B20	IRQ20	
PCA	B21	B21	IRQ21	
WDT	B22	B22	IRQ22	
RTC	B23	B23	IRQ23	
				IPR5
ADC	B24	B24	IRQ24	
-	B25	B25	IRQ25	
VC1	B26	B26	IRQ26	
VC2	B27	B27	IRQ27	
				IPR6
LVD	B28	B28	IRQ28	
-	B29	B29	IRQ29	
EFCTRL	RAMCTRL	B30	IRQ30	
	CLKTRIM	B31	IRQ31	
				IPR7

NVIC

Figure 5-7 Interrupt structure diagram

The interrupt structure diagram of this system is shown in Figure 5-7 . A few things to note:

The respective interrupt enable of the peripheral interrupt source is not marked in the figure, here only the interrupt signal after the peripheral interrupt i No. Logical block diagram.

IRQ30 has 2 peripheral interrupt input multiplexing, and the interrupt flag bits of these 2 peripherals must be read separately to determine which one is Interrupts of two peripherals.

If the peripheral interrupt source has a high level, no matter whether the NVIC interrupt enable register SCS_SETENA is set or not,

The interrupt pending register SCS_SEPEND will be set, indicating that the corresponding peripheral interrupt source has an interrupt.

Only when the interrupt enable register SCS_SETENA is set, the corresponding interrupt IRQ will be sent to the processor, and the execution

The corresponding interrupt program.

The high level interrupt signal of the peripheral interrupt source must be cleared in the interrupt program, the interrupt pending register SCS_SETPEND Automatically cleared by hardware.

The interrupt priority register SCS_IPR0- SCS_IPR7 sets the priority of 32 interrupt sources, 00 has the highest priority

High, 11 has the lowest priority. When the priority is the same, the priority is determined by the interrupt number, the smaller the number, the higher th

Page 77

5.8 Register

Base address: 0xE000 E000

register	Offset address	describe
SCS_SETENA	0x100	Interrupt request enable register
SCS_CLRENA	0x180	Interrupt request clear enable register
SCS_SETPEND	0x200	Interrupt set pending register
SCS_CLRPEND	0x280	Interrupt clear pending register
SCS_IPR0	0x400	Interrupt #0-Interrupt #3 priority register
SCS_IPR1	0x404	Interrupt #4-Interrupt #7 Priority Register
SCS_IPR2	0x408	Interrupt #8-Interrupt #11 Priority Register
SCS_IPR3	0x40C	Interrupt #12-Interrupt #15 priority register
SCS_IPR4	0x410	Interrupt #16-Interrupt #19 Priority Register
SCS_IPR5	0x414	Interrupt #20-Interrupt #23 priority register
SCS_IPR6	0x418	Interrupt #24-Interrupt #27 Priority Register
SCS_IPR7	0x41C	Interrupt #28-Interrupt #31 Priority Register
SCS_PRIMASK	-	Interrupt mask special register

5.8.1 Interrupt enable setting register (SCS_SETENA)

Offset address: 0x100

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16
SETENA[31:16]														
RW														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
SETENA[15:0]														
RW														

Bit	mark	Function description
-----	------	----------------------

31:0	SETENA	Set enable interrupt #0 to interrupt #31; write "1" to set the bit, write "0" to invalid
[31:0]	[0]:IRQ0	
	[1]:IRQ1	
	[2]:IRQ2	
	...	
	[31]:IRQ31	

Page 78**5.8.2 Interrupt enable clear register (SCS_CLRENA)**

Offset address: 0x180

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four twenty three twenty two twenty one	20	19	18	17	16	
CLRENA													
RW													
15	14	13	12	11	10	9	8	7	6	5	4	3	2
CLRENA													
RW													

Bit	mark	describe
31:0	CLRENA	Clear enable interrupt #0 to interrupt #31; write "1" to clear, write "0" to invalid

5.8.3 Interrupt pending status setting register (SCS_SETPEND)

Offset address: 0x200

Reset value: 0x0000 0000

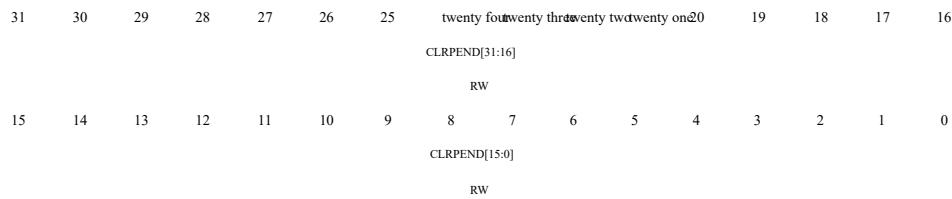
31	30	29	28	27	26	25	twenty four twenty three twenty two twenty one	20	19	18	17	16	
SETPEND[31:16]													
RW													
15	14	13	12	11	10	9	8	7	6	5	4	3	2
SETPEND[15:0]													
RW													

Bit	mark	Function description
31:0	SETPEND	Set the suspended state of interrupt #0 to interrupt #31; write "1" to set the bit, write "0" to invalid
	[0]:IRQ0	
	[1]:IRQ1	
	[2]:IRQ2	
	...	
	[31]:IRQ31	

Page 79**5.8.4 Interrupt pending status clear register (SCS_CLRPEND)**

Offset address: 0x280

Reset value: 0x0000 0000



Bit	mark	describe
31:0	CLRPEND	Clear the suspended state of interrupt #0 to interrupt #31; write "1" to clear zero, write "0" to invalid
	[0]:IRQ0	
	[1]:IRQ1	
	[2]:IRQ2	
	...	
	[31]:IRQ31	

Page 80

5.8.5 Interrupt Priority Register (SCS_IPR0)

Offset address: 0x400

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR0[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR0[15:0]															
RW															

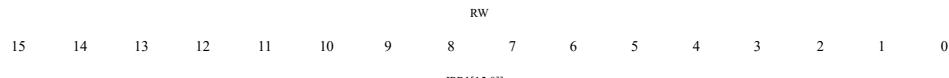
Bit	mark	Function description
31:0	IPR0[31:0]	The priority of interrupt #0 to interrupt #3; [31:30]: Priority of interrupt #3 [23:22]: Interrupt #2 priority [15:14]: Interrupt #1 priority [7:6]: Priority of interrupt #0
		Among them, 00 has the highest priority and 11 has the lowest priority

5.8.6 Interrupt Priority Register (SCS_IPR1)

Offset address: 0x404

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR1[31:16]															
IPR1[15:0]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR1[15:0]															



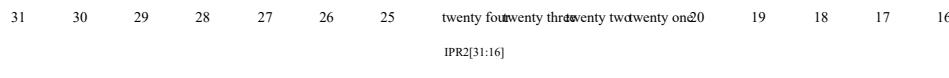
RW

Bit	mark	Function description
31:0	IPR1[31:0]	The priority of interrupt #4 to interrupt #7; [31:30]: Priority of interrupt #7 [23:22]: Interrupt #6 priority [15:14]: Interrupt #5 priority [7:6]: Priority of interrupt #4
		Among them, 00 has the highest priority and 11 has the lowest priority

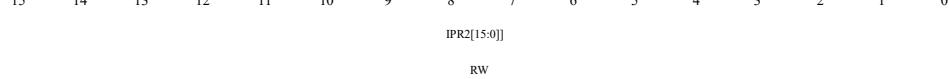
Page 82**5.8.7 Interrupt Priority Register (SCS_IPR2)**

Offset address: 0x408

Reset value: 0x0000 0000



RW



RW

Bit	mark	Function description
31:0	IPR2[31:0]	The priority of interrupt #8 to interrupt #11;

[31:30]: Priority of interrupt #11

[23:22]: Priority of interrupt #10

[15:14]: Interrupt #9 priority

[7:6]: Priority of interrupt #8

Among them, 00 has the highest priority and 11 has the lowest priority

Page 83

5.8.8 Interrupt Priority Register (SCS_IPR3)

Offset address: 0x40C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR3[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR3[15:0]															
RW															

Bit	mark	Function description
-----	------	----------------------

31:0 IPR3[31:0] The priority of interrupt #12 to interrupt #15;

[31:30]: Priority of interrupt #15

[23:22]: Priority of interrupt #14

[15:14]: Interrupt #13 priority

[7:6]: Priority of interrupt #12

Among them, 00 has the highest priority and 11 has the lowest priority

Page 84**5.8.9 Interrupt Priority Register (SCS_IPR4)**

Offset address: 0x410

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR4[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR4[15:0]															
RW															

Bit	mark	Function description
31:0	IPR4[31:0]	The priority of interrupt #16 to interrupt #19; [31:30]: Priority of interrupt #19 [23:22]: Interrupt #18 priority [15:14]: Priority of interrupt #17 [7:6]: Priority of interrupt #16 Among them, 00 has the highest priority and 11 has the lowest priority

Page 85**5.8.10 Interrupt Priority Register (SCS_IPR5)**

Offset address: 0x414

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR5[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR5[15:0]															
RW															

Bit	mark	Function description
31:0	IPR5[31:0]	The priority of interrupt #20 to interrupt #23; [31:30]: Priority of interrupt #23 [23:22]: Priority of interrupt #22 [15:14]: Priority of interrupt #21 [7:6]: Priority of interrupt #20
		Among them, 00 has the highest priority and 11 has the lowest priority

Page 86**5.8.11 Interrupt Priority Register (SCS_IPR6)**

Offset address: 0x418

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
IPR6[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR6[15:0]															
RW															

Bit	mark	Function description
31:0	IPR6[31:0]	The priority of interrupt #24 to interrupt #27; [31:30]: Priority of interrupt #27 [23:22]: Interrupt #26 priority [15:14]: Interrupt #25 priority [7:6]: Priority of interrupt #24
		Among them, 00 has the highest priority and 11 has the lowest priority

Page 87

5.8.12 Interrupt Priority Register (SCS_IPR7)

Offset address: 0x41C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR7[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR7[15:0]															
RW															

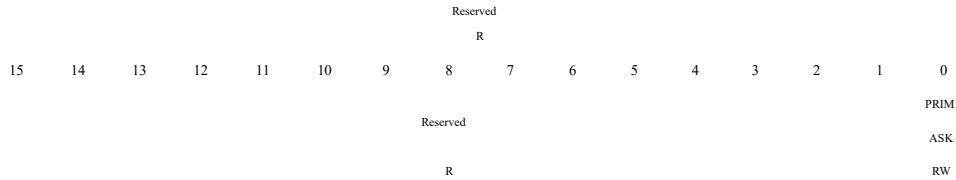
Bit	mark	Function description
31:0	IPR7[31:0]	The priority of interrupt #28 to interrupt #31; [31:30]: Priority of interrupt #31 [23:22]: Priority of interrupt #30 [15:14]: Priority of interrupt #29 [7:6]: Priority of interrupt #28
		Among them, 00 has the highest priority and 11 has the lowest priority

5.8.13 Interrupt mask special register (SCS_PRIMASK)

Offset address: ---

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IPR7[31:16]															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IPR7[15:0]															
RW															

**SCS_PRIMASK**

Reset value: 0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															
RO															
BIT	symbol	describe													
31:1	Reserved														
0	PRIMASK	<p>After setting, all interrupts except NMI and hardware error exceptions will be masked</p> <p>After clearing, all exceptions and interrupts will not be masked</p> <p>This special register needs to be accessed through the MSR and MRS special register operation instructions, or</p> <p>Use the change processor state instruction CPS to access. When dealing with time-sensitive applications, operations are required</p> <p>PRIMASK register.</p>													

5.9 Basic software operation

5.9.1 External interrupt enable

Each peripheral module has its own interrupt enable register. When an interrupt operation is required, it must first Turn on the peripheral's own interrupt enable. The operation of the enable bit is not discussed in this chapter, please refer to the respective peripheral mo Chapter description.

5.9.2 NVIC interrupt enable and clear enable

Cortex-M0+ processor supports up to 32 interrupt sources, and each interrupt source corresponds to an interrupt enable bit and cleared Enable bit. So there is a 32-bit interrupt enable register SCS_SETENA and a 32-bit clear enable register

SCS_CLRENA. If you want to enable an interrupt, set the corresponding bit of the SCS_SETENA register to 1. like If you want to clear an interrupt, set the corresponding bit in the SCS_CLRENA register.

Note that the interrupt enable mentioned here is only for the processor NVIC, and the interrupt generation of each peripheral is related to No, it is determined by the interrupt control register of the peripheral, and has nothing to do with SCS_SETENA and SCS_CLRENA.

5.9.3 NVIC interrupt suspension and clear suspension

If an interrupt occurs but cannot be processed immediately, the interrupt request will be suspended. The suspended state is saved in one In this register, if the current priority of the processor has not been lowered to handle the pending request, and there is no hand The suspended state is automatically cleared, and the state will always remain legal.

When the processor starts to enter the interrupt processing, the hardware will automatically cause the suspend state to be cleared.

You can set pending SCS_SETPEND and interrupt clear pending SCS_CLRPEND by operating interrupts

Register to access or modify interrupt pending status. The interrupt pending status register allows software to trigger interrupts.

5.9.4 NVIC interrupt priority

Set the SCS_IPR0- SCS_IPR7 register to determine the priority of SCS_IRQ0- SCS_IRQ32. Interrupt priority

The programming of the level register should be done before the interrupt is enabled, which is usually completed at the beginning of the program. Shoul

Change the interrupt priority after enabling. The result of this situation is unpredictable and is not supported by the Cortex-M0+ processor. hold.

5.9.5 NVIC interrupt mask

Some time-sensitive applications need to disable all interrupts within a short period of time. You can use the interrupt mask to register

The register SCS_PRIMASK is implemented. The special register SCS_PRIMASK has only 1 bit valid, and it is reset

The latter defaults to 0. When this register is 0, all interrupts and exceptions are allowed; and when it is set to 1, only NMI (not supported by this system) and hardware error exceptions are enabled. In fact, when SCS_PRIMASK is set to After 1, the current priority of the processor drops to 0 (the highest priority of the summable value).

You can program the SCS_PRIMASK register in a variety of ways, using assembly language, you can use the MS R instruction

When setting and clearing the SCS_PRIMASK register. If you use C language and CMSIS device driver library, the user

You can use the following functions to set and clear PRIMASK.

```
void __enable_irq(void); //Clear PRIMASK
void __disable_irq(void); //Set PRIMASK
```

Page 91

6 Port controller (GPIO)

6.1 Introduction to Port Controller

This product has 16 digital general-purpose input and output ports P01-P03, P14-P15, P23-P27, P31-P36 and 1 number

Word general-purpose input port P00. Analog signal ADC/VC/LVD input and output signals, various functional modules (such as SPI,

The input and output signals of UART, I2C, Timer, etc. and the input and output signals for testing and debugging functions can all be combined with the

Word port multiplexing.

Each port can be configured as an internal pull up/pull down input, high impedance input (floating

input), push-pull output (CMOS output), open drain output (open drain output), two-speed drive capability output. for

To prevent abnormal actions of external devices when the chip is abnormally reset, the port is configured as a high-impedance input after the chip is rese

In order to avoid leakage caused by high-impedance input, the user must configure the port accordingly (configuration

Into an internal pull-up/pull-down input or output).

After the digital port is configured as an analog port, the digital function is isolated, and the digital "1" and "0" cannot be output. The CPU read terminal

The result of the mouth is "0".

All ports can provide external interrupts, and each interrupt can be configured as high-level trigger, low-level trigger,

There are 4 types of rising edge triggering and falling edge triggering. You can check the interrupt flag bit of Px_STAT[n] to find out the corresponding

Disconnect the trigger port. In addition, the interrupt of each port can wake up the chip from sleep mode/deep sleep mode to work

model.

Page 92

6.2 Main Features of Port Controller

The port controller supports the following features:

Port multiplexing function

- Analog function pin multiplexing
- Debug pin multiplexing
- Digital general-purpose pin multiplexing
- Digital function pin multiplexing

Configuration function

- Support pull-up/pull-down
- Low drive/high drive
- Push-pull output
- Open drain output

External interrupt source

- High level/low level
- Rising edge/falling edge

Support interrupt in working mode/sleep mode/deep sleep mode

6.3 Port controller function description

6.3.1 Port configuration function

Each port can be configured as an analog port or a digital port through the configuration register (PxADS) according to the system requirements.

Word port. When configured as a digital port, you can also configure the corresponding registers to achieve the following features:

1. Internal pull-up (PxPU)/pull-down (PxPD)

The pull-up register (PxPU) and the pull-down register (PxPD) correspond to the port pull-up enable and port pull-down enable respectively,

When the corresponding bit is '1', set the corresponding bit pin pull-up/pull-down enable, when it is '0', the corresponding bit pin is prohibited

Pull up/down.

2. Two-speed drive output (PxDR)

The drive capability can be changed through the PxDR register. When PxDR is '1', it is low drive capability, and PxDR is '0'

When the drive capacity is high.

3. Open drain output (PxOD)

Set the pin output status through the PxOD register. When PxOD is '1', the port open-drain output is enabled, which is

When '0', the port open-drain output is disabled. When the open-drain pin is not connected to an external pull-up resistor, it can only output low level

If you need to have the function of outputting a high level at the same time, you need a pull-up resistor.

4. Direction selection (PxDIR)

Used to set the direction of the port pins. When PxDIR is '0', the port is output, when PxDIR is '1'

The candidate port is input.

5. Output high and low level selection (PxOUT), which can be accessed through the AHB bus

When the port pin is configured as output, if PxOUT is '1', the port pin output is high level, if configured as

Open-drain output requires an external pull-up resistor to pull it high. If PxOUT is '0', it outputs low level.

6. The input level status (PxIN) can be accessed through the AHB bus

The pin level after synchronization can be obtained by reading the PxIN register. When PxIN is '1', it is high level, and PxIN

Low level when it is '0'

Note: The above features are invalid when configured as an analog port.

The relationship between port status and register configuration is as follows:

IO status	IO direction	PxADS	PxDIR	PxOUT	PxIN	PxPU	PxPD	PxOD	PxDR	Px_SEL
simulation	input Output	1	W	W	0	W	W	W	W	W
Floating	enter	0	1	W	X	0	0	W	W	0
drop down	enter	0	1	W	0	0	1	W	W	0
pull up	enter	0	1	W	1	1	0	W	W	0
pull up	enter	0	1	W	1	1	1	W	W	0
1	enter	0	1	W	1	W	W	W	W	0
0	enter	0	1	W	0	W	W	W	W	0
1	Output	0	0	1	1	W	W	0	W	0
0	Output	0	0	0	0	W	W	0	W	0
1	Output	0	0	W	1	W	W	0	W	0
0	Output	0	0	W	0	W	W	0	W	0
(SET)1/(CLR)0	Output	0	0	W	(SET)1/(CLR)0	W	W	0	W	0
0	Output	0	0	0	0	W	W	1	W	0
Z	Output	0	0	1	X	0	0	1	W	0
0	Output	0	0	1	0	0	1	1	W	0
1	Output	0	0	1	1	1	0	1	W	0
1	Output	0	0	1	1	1	1	1	W	0

Note: 0-Logic low

1-Logic high

W-Whatever 0 or 1

X-unknown state

Z-high impedance

Table 6-1 Truth Table of Port Status

The port circuit structure is shown in the figure below:



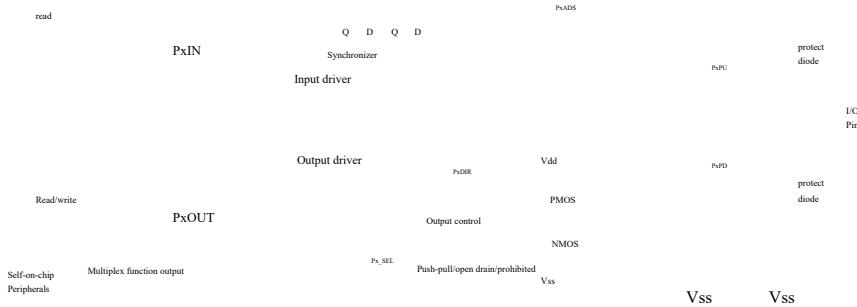


Figure 6-1 Schematic diagram of port circuit

6.3.2 Port writing

Port input value/output value registers (PxIN/PxOUT) support AHB bus read and write. For the AHB bus, the system

The processing cycle of the system clock (HCLK) is different from that of other buses. Every two HCLK cycles, IO flips by one.

Second-rate. The following figure shows the fastest timing of the AHB bus port flip:



Figure 6-2 AHB bus port changes with the system clock

6.3.3 Port reading

Each port can obtain the port pin level by reading the PxIN register. The bits of the PxIN register are

The latch in front of it forms a synchronizer, thus avoiding a short period of time when the state of the system clock changes.

The signal is unstable due to pin level changes, but it also introduces a delay. The same as reading port pin data

The step diagram is as follows:



PIN voltage

PxIN

T_{pd,max}
T_{pd,min}

Figure 6-3 Read port pin data synchronization diagram

In the clock period after the rising edge of the system clock, the pin level signal will be locked in the internal register, such as the shaded part As shown, after the next rising edge of the system clock, a stable pin level signal can be read. And then the system On the rising edge of the clock, the data is latched into the PxIN register. The signal conversion delay Tpd is 1-2 system clocks.

Notice:

-Handling of unconnected pins:

If there are unconnected pins during use, in order to avoid the pin not being determined in other digital input enable modes The level causes the floating current consumption. It is recommended to assign a certain level to the pin.

6.3.4 Port multiplexing function

Port multiplexing is one of the main functions of the port controller. Through the configuration register, the port can be flexibly configured as a module Quasi port/test debugging port/digital universal port/digital function port.

The PxADS register is used for digital port/analog port switching. When PxADS is '1', the port is configured as an analog port

At this time, the digital function is isolated, and the numbers "1" and "0" cannot be output. The result of the CPU reading the port is "0". when

When PxADS is '0', the port is configured as a digital port. At this time, the digital universal port is realized by configuring the Px_sel register

Port/digital function port switch, each port can be independently configured as a function port required by the system. Test the debug side

For port configuration information, please refer to the relevant chapters.

simulation		number							
PxADS=0									
PxADS=1		Px_sel=0	Px_sel=1	Px_sel=2	Px_sel=3	Px_sel=4	Px_sel=5	Px_sel=6	Px_sel=7
AIN4	VCIN4	P34	PCA_CH0	LPUART_TXD	TIM5_CHA	TIM0_EXT	TIM4_CHA	RTC_IHZ	TIM1_TOG
AIN5	VCIN5	P35	UART1_RXD	TIM6_CHB	UART0_RXD	TIM0_GATE	TIM4_CHB	SPI_MISO	I2C_SDA
AIN6	VCIN6	P36	UART1_RXD	TIM6_CHA	UART0_RXD	PCA_CH4	TIM5_CHA	SPI_MOSI	I2C_SCL
ADC_VREF									
AIN7	VCIN7	P01	UART0_RXD	I2C_SDA	UART1_RXD	TIM0_TOG	TIM5_CHB	SPI_SCK	TIM2_EXT
XTHI									
AIN8		P02	UART0_RXD	I2C_SCL	UART1_RXD	TIM0_TOGN	TIM6_CHA	SPI_CS	TIM2_GATE

XTHO								
LVDIN1	P03	PCA_CH3	SPI_CS	TIM6_CHB	LPTIM_EXT	RTC_IHz	PCA_ECI	VCO_OUT
XTLO	P15	I2C_SDA	TIM2_TOG	TIM4_CHB	LPTIM_GATE	SPI_SCK	UART0_RXD	LVD_OUT
XTLI	P14	I2C_SCL	TIM2_TOGN	PCA_ECI	ADC_RDY	SPI_CS	UART0_TXD	NC
LVDIN2	VCINO	P23	TIM6_CHA	TIM4_CHB	TIM4_CHA	PCA_CH0	SPI_MISO	UART1_TXD
AIN0	P24	TIM4_CHB	TIM5_CHB	HCLK_OUT	PCA_CH1	SPI_MOSI	UART1_RXD	VC1_OUT
LVDIN3	VCINI	P25	SPI_SCK	PCA_CH0	TIM5_CHA	LVD_OUT	LPUART_RXD	I2C_SDA
AIN1	P26	SPI_MOSI	TIM4_CHA	TIM5_CHB	PCA_CH2	LPUART_TXD	I2C_SCL	TIM1_EXT
	P27/SWDIO	SPI_MISO	TIM5_CHA	TIM6_CHA	PCA_CH3	UART0_RXD	RCH_OUT	XTH_OUT
	P31/SWCLK	LPTIM_TOG	PCA_ECI	PCLK_OUT	VCO_OUT	UART0_TXD	RCL_OUT	HCLK_OUT
AIN2	VCIN2	P32	LPTIM_TOGN	PCA_CH2	TIM6_CHB	VCI_OUT	UART1_TXD	PCA_CH4
AIN3	VCIN3	P33	LPUART_RXD	PCA_CH1	TIM5_CHB	PCA_ECI	UART1_RXD	XTL_OUT
	P00							TIM1_TOGN
	Reset							

Table 6-2 Port reuse table

6.3.5 Port interrupt function

Every digital universal port can be interrupted by an external signal source, the external signal source can be high level/low level/

Rising edge/falling edge 4 types of signals, the corresponding interrupt enable register is high level interrupt enable register/

Low level interrupt enable register/rising edge interrupt enable register/falling edge interrupt enable register.

When an interrupt is triggered, you can determine which port triggered the interrupt by querying the interrupt status register.

Zero interrupt clear register can clear the corresponding interrupt status flag bit.

Page 99

6.4 Port configuration operation

6.4.1 Port multiplexing operation process

Port reuse configuration as analog port

Step1: Set the register Px_ADS to 1

Port multiplexing is configured as a digital universal port

- a) Set register Px_ADS to 0
- b) Set the register Px_sel to 0
- c) Set the register PxDIR to 1: The port direction is input, and the CPU can read the port status PxIN
- d) Set the register PxDIR to 0: the port direction is output
- e) Set register PxOUT to 1: Port output high level
- f) Set the register PxOUT to 0: the port outputs low level

Port multiplexing is configured as a digital function port

- a) Set register Px_ADS to 0
- b) Set the register Px_sel to 1~7 (according to the system requirements, refer to the port multiplexing table)
- c) Set the register PxDIR (according to system requirements)

Port reuse configuration as debug test port

Refer to related chapters of test and debugging.

Port multiplexing configuration as infrared output signal

Port P23 can be configured as an infrared output signal with a frequency of 38K.

- a) Set register P23_ADS to 0
- b) Set register P23_sel to 7
- c) Set the register P2DIR[3] to 0: the port direction is output
- d) Set bit14 of register GPIO_CTRL1 to select infrared signal output polarity
- e) Set the register P2OUT[3] to control the output of the gated infrared signal

Page 100

6.4.2 Port interrupt operation flow

High level interrupt

- a) Set register Px_ADS to 0
- b) Set the register Px_sel to 0
- c) Set the register PxDIR to 1
- d) Set the register PxHIE to 1
- e) Read the interrupt status register Px_STAT after the interrupt is triggered
- f) Set the register Px_ICLR to 0 and clear the interrupt status register Px_STAT

Low level interrupt

- a) Set register Px_ADS to 0
- b) Set the register Px_sel to 0
- c) Set the register PxDIR to 1
- d) Set the register PxLIE to 1
- e) Read the interrupt status register Px_STAT after the interrupt is triggered
- f) Set the register Px_ICLR to 0 and clear the interrupt status register Px_STAT

Interrupt on rising edge

- a) Set register Px_ADS to 0
- b) Set the register Px_sel to 0
- c) Set the register PxDIR to 1
- d) Set the register PxRIE to 1
- e) Read the interrupt status register Px_STAT after the interrupt is triggered
- f) Set the register Px_ICLR to 0 and clear the interrupt status register Px_STAT

Falling edge interrupt

- a) Set register Px_ADS to 0
- b) Set the register Px_sel to 0
- c) Set the register PxDIR to 1
- d) Set the register PxFIE to 1
- e) Read the interrupt status register Px_STAT after the interrupt is triggered

f) Set the register Px_ICLR to 0 and clear the interrupt status register Px_STAT

6.4.3 Port configuration operation process

Pull-up enable

a) Set the register PxPU to 1

Pull-down enable

a) Set the register PxPU to 0

b) Set the register PxPD to 1

High driving ability

a) Set register PxDR to 0

Open drain output

a) Set the register PxOD to 1

6.5 Port Controller Register Description

Register list

Base address: 0x40020C00

Offset register name	access	Register description
----------------------	--------	----------------------

0x04	P01_SEL	RW	Port P01 function configuration register
0x08	P02_SEL	RW	Port P02 function configuration register
0x0c	P03_SEL	RW	Port P03 function configuration register
0x50	P14_SEL	RW	Port P14 function configuration register
0x54	P15_SEL	RW	Port P15 function configuration register
0x8c	P23_SEL	RW	Port P23 function configuration register
0x90	P24_SEL	RW	Port P24 function configuration register
0x94	P25_SEL	RW	Port P25 function configuration register
0x98	P26_SEL	RW	Port P26 function configuration register
0x9c	P27_SEL	RW	Port P27 function configuration register
0xc4	P31_SEL	RW	Port P31 function configuration register
0xc8	P32_SEL	RW	Port P32 function configuration register
0xcc	P33_SEL	RW	Port P33 function configuration register
0xd0	P34_SEL	RW	Port P34 function configuration register
0xd4	P35_SEL	RW	Port P35 function configuration register
0xd8	P36_SEL	RW	Port P36 function configuration register
0x100	P0DIR	RW	Port P0 input and output configuration register
0x104	P0IN	RO	Port P0 input value register
0x108	P0OUT	RW	Port P0 output value configuration register
0x10c	P0ADS	RW	Port P0 digital-analog configuration register
0x11c	P0DR	RW	Port P0 drive capability configuration register
0x120	P0PU	RW	Port P0 pull-up enable configuration register
0x124	P0PD	RW	Port P0 pull-down enable configuration register
0x12c	P0OD	RW	Port P0 open-drain output configuration register

0x130	P0HIE	RW	Port P0 high level interrupt enable configuration register
0x134	P0LIE	RW	Port P0 low-level interrupt enable configuration register
0x138	P0RIE	RW	Port P0 rising edge interrupt enable configuration register
0x13c	P0FIE	RW	Port P0 falling edge interrupt enable configuration register
0x200	P0_STAT	RO	Port P0 interrupt status register
0x210	P0_ICLR	RW	Port P0 interrupt clear register
0x140	P1DIR	RW	Port P1 input and output configuration register
0x144	P1IN	RO	Port P1 input value register
0x148	P1OUT	RW	Port P1 output value configuration register

0x14c	P1ADS	RW	Port P1 digital-analog configuration register
0x15c	P1DR	RW	Port P1 drive capability configuration register
0x160	P1PU	RW	Port P1 pull-up enable configuration register
0x164	P1PD	RW	Port P1 pull-down enable configuration register
0x16c	P1OD	RW	Port P1 open-drain output configuration register
0x170	P1HIE	RW	Port P1 high level interrupt enable configuration register
0x174	P1LIE	RW	Port P1 low-level interrupt enable configuration register
0x178	P1RIE	RW	Port P1 rising edge interrupt enable configuration register
0x17c	P1FIE	RW	Port P1 falling edge interrupt enable configuration register
0x240	P1_STAT	RO	Port P1 interrupt status register
0x250	P1_ICLR	RW	Port P1 interrupt clear register
0x180	P2DIR	RW	Port P2 input and output configuration register
0x184	P2IN	RO	Port P2 input value register
0x188	P2OUT	RW	Port P2 output value configuration register
0x18c	P2ADS	RW	Port P2 digital-analog configuration register
0x19c	P2DR	RW	Port P2 drive capability configuration register
0x1a0	P2PU	RW	Port P2 pull-up enable configuration register
0x1a4	P2PD	RW	Port P2 pull-down enable configuration register

Page 104

0x1ac	P2OD	RW	Port P2 open-drain output configuration register
0x1b0	P2HIE	RW	Port P2 high level interrupt enable configuration register
0x1b4	P2LIE	RW	Port P2 low-level interrupt enable configuration register
0x1b8	P2RIE	RW	Port P2 rising edge interrupt enable configuration register
0x1bc	P2FIE	RW	Port P2 falling edge interrupt enable configuration register
0x280	P2_STAT	RO	Port P2 interrupt status register
0x290	P2_ICLR	RW	Port P2 interrupt clear register
0x1c0	P3DIR	RW	Port P3 input and output configuration register
0x1c4	P3IN	RO	Port P3 input value register
0x1c8	P3OUT	RW	Port P3 output value configuration register
0x1cc	P3ADS	RW	Port P3 digital-analog configuration register
0x1dc	P3DR	RW	Port P3 drive capability configuration register
0x1e0	P3PU	RW	Port P3 pull-up enable configuration register
0x1e4	P3PD	RW	Port P3 pull-down enable configuration register
0x1ec	P3OD	RW	Port P3 open-drain output configuration register

0x1f0	P3HIE	RW	Port P3 high level interrupt enable configuration register
0x1f4	P3LIE	RW	Port P3 low-level interrupt enable configuration register
0x1f8	P3RIE	RW	Port P3 rising edge interrupt enable configuration register
0x1fc	P3FIE	RW	Port P3 falling edge interrupt enable configuration register
0x2c0	P3_STAT	RO	Port P3 interrupt status register
0x2d0	P3_ICLR	RW	Port P3 interrupt clear register
0x304	GPIO_CTRL1	RW	Port auxiliary function configuration register 1
0x308	GPIO_CTRL2	RW	Port auxiliary function configuration register 2
0x30c	GPIO_CTRL3	RW	Port auxiliary function configuration register 3
0x310	GPIO_CTRL4	RW	Port auxiliary function configuration register 4

Page 105

6.5.1 Port P0

6.5.1.1 Port P01 Function Configuration Register (P01_SEL)

Offset address: 0x04

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16	Reserved	P01_sel
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	Reserved	RW

Bit mark Function description

31:3 Reserved

2:0 P01_sel Port P01 function selection.

000: GPIO P01

001: UART0_RXD RXD signal of UART0 module

010: I2C_SDA I2C module data signal

011: UART1_TXD UART1 module TXD signal

100: TIM0_TOG Timer0 module flip signal

101: TIM5_CHB Advanced Timer module channel 1 B signal

110: SPI_SCK SPI module clock signal

111: TIM2_EXT Timer2 module external clock input signal

Page 106**6.5.1.2 Port P02 Function Configuration Register (P02_SEL)**

Offset address: 0x08

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16	Reserved	P02_sel
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	Reserved	RW

Bit mark Function description

31:3 Reserved

2:0 P02_sel Port P02 function selection.

000: GPIO P02

001: UART0_TXD UART0 module TXD signal

010: I2C_SCL I2C module clock signal

011: UART1_RXD RXD signal of UART1 module

100: TIM0_TOGN Timer0 module flip signal reverse signal

101: TIM6_CHA Advanced Timer module channel 2 A signal

110: SPI_CS SP I module host mode chip select signal

111: TIM2_GATE Timer2 module gate control signal

Page 107

6.5.1.3 Port P03 Function Configuration Register (P03_SEL)

Offset address: 0x0C

Reset value: 0x0000 0000

Reserved

15 14 13 12

11

8

2 1

1

Bit	mark	Function description
31:3	Reserved	
2:0	P03_sel	Port P03 function selection.
	000: GPIO_P03	
	001: PCA_CH3	PCA module channel 3 capture/compare signal
	010: SPI_CS	SPI module host mode chip select signal
	011: TIM6_CHB	Advanced Timer module channel 2 B signal
	100: LPTIM_EXT	Timer3 module external clock input signal
	101: RTC_1Hz	RTC module 1Hz output signal
	110: PCA_ECI	PCA module external clock input signal
	111: VC0_OUT	VC0 module output

6.5.1.4 Port P0 Input and Output Configuration Register (P0DIR)

Offset address: 0x100

Reset value: 0xffff ffff

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved																			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	P0D	P0D	P0D	Res
																IR3	IR2	IR1	
																RW	RW	RW	

Bit mark Function description

31:4 Reserved

3 P0DIR3 Port P03 Input and Output Configuration Register

1: Configure as input

0: Configure as output

2 P0DIR2 Port P02 Input and Output Configuration Register

1: Configure as input

0: Configure as output

1 P0DIR1 Port P01 Input and Output Configuration Register

1: Configure as input

0: Configure as output

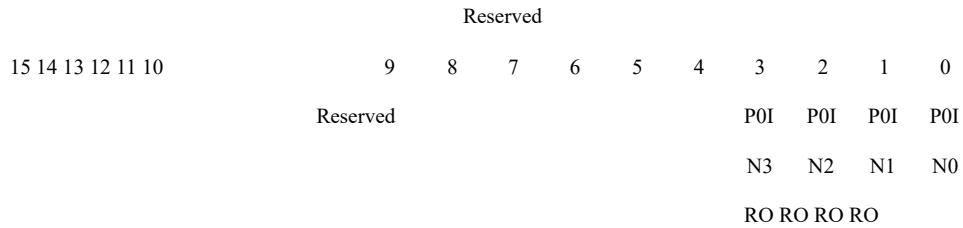
0 Reserved

6.5.1.5 Port P0 Input Value Register (P0IN)

Offset address: 0x104

Reset value: NA

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16



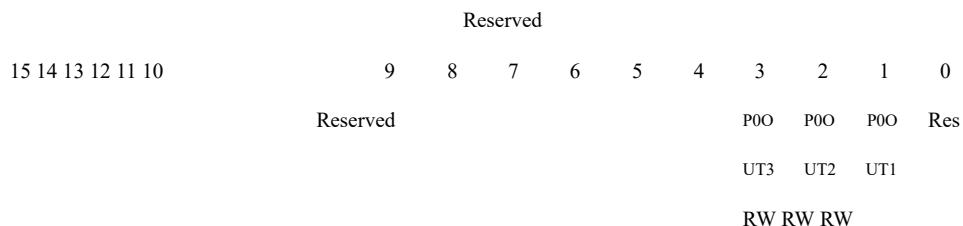
Bit	mark	Function description
31:4	Reserved	
3	P0IN3	Port P03 input value register 1: Input is high level 0: Input is low level
2	P0IN2	Port P02 input value register 1: Input is high level 0: Input is low level
1	P0IN1	Port P01 input value register 1: Input is high level 0: Input is low level
0	P0IN0	Port P00 input value register 1: Input is high level 0: Input is low level

6.5.1.6 Port **P0** output value configuration register (**P0OUT**)

Offset address: 0x108

Reset value: NA

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16



Bit	mark	Function description
31:4	Reserved	
3	P0OUT3 Port P03 output value configuration register	<p>1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.</p> <p>0: Output low level.</p>
2	P0OUT2 port P02 output value configuration register	<p>1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.</p> <p>0: Output low level.</p>
1	P0OUT1 port P01 output value configuration register	<p>1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.</p> <p>0: Output low level.</p>
0	Reserved	

Page 111**6.5.1.7 Port P0 Digital-to-Analog Configuration Register (P0ADS)**

Offset address: 0x10C

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved										5	4	3	2	1	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						Reserved				P0A	P0A	P0A	Res		
										DS3	DS2	DS1			
										RW	RW	RW			

Bit	mark	Function description
31:4	Reserved	
3	P0ADS3 Port P03 Digital-to-Analog Configuration Register	<p>1: Configure as an analog port</p>

	0: Configure as a digital port
2	P0ADS2 port P02 digital-analog configuration register
	1: Configure as an analog port
	0: Configure as a digital port
1	P0ADS1 port P01 digital-analog configuration register
	1: Configure as an analog port
	0: Configure as a digital port
0	Reserved

Page 112**6.5.1.8 Port P0 Drive Capability Configuration Register (P0DR)**

Offset address: 0x11C

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						Reserved				P0D	P0D	P0D	Res		
										R3	R2	R1			
										RW	RW	RW			

Bit	mark	Function description
31:4	Reserved	
3	P0DR3	Port P03 drive capability configuration register
		1: Low drive capability
		0: High drive capability
2	P0DR2	Port P02 drive capability configuration register
		1: Low drive capability
		0: High drive capability
1	P0DR1	Port P01 drive capability configuration register
		1: Low drive capability

0: High drive capability

0 Reserved

Page 113**6.5.1.9 Port P0 pull-up enable configuration register (P0PU)**

Offset address: 0x120

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15 14 13 12 11 10	9 8 7 6 5 4 3 2 1 0	P0P P0P P0P Res
	Reserved	U3 U2 U1
		RW RW RW

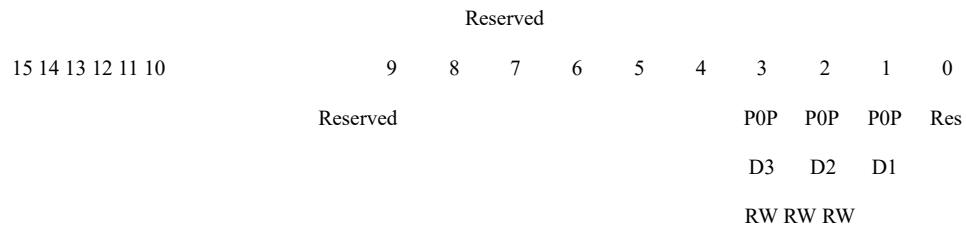
Bit	mark	Function description
31:4	Reserved	
3	P0PU3	Port P03 pull-up enable configuration register 1: enable 0: prohibited
2	P0PU2	Port P02 pull-up enable configuration register 1: enable 0: prohibited
1	P0PU1	Port P01 pull-up enable configuration register 1: enable 0: prohibited
0	Reserved	

Page 114**6.5.1.10 Port P0 pull-down enable configuration register (P0PD)**

Offset address: 0x124

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16



Bit	mark	Function description
31:4	Reserved	
3	P0PD3	Port P03 pull-down enable configuration register 1: enable 0: prohibited
2	P0PD2	Port P02 pull-down enable configuration register 1: enable 0: prohibited
1	P0PD1	Port P01 pull-down enable configuration register 1: enable 0: prohibited
0	Reserved	

6.5.1.11 Port P0 open-drain output configuration register (P0OD)

Offset address: 0x12C

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

										Reserved					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										P0O	P0O	P0O	Res		
										D3	D2	D1			
										RW	RW	RW			

Bit	mark	Function description
31:4	Reserved	
3	P0OD3	Port P03 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
2	P0OD2	Port P02 open drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
1	P0OD1	Port P01 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
0	Reserved	

6.5.1.12 Port P0 high level interrupt enable configuration register (P0HIE)

Offset address: 0x130

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Bit	mark	Function description
31:4	Reserved	
3	P0HIE3 Port P03 High Level Interrupt Enable Configuration Register	
	1: enable	
	0: prohibited	
2	P0HIE2 Port P02 High Level Interrupt Enable Configuration Register	
	1: enable	
	0: prohibited	
1	P0HIE1 port P01 high level interrupt enable configuration register	
	1: enable	
	0: prohibited	
0	P0HIE0 port P00 high level interrupt enable configuration register	
	1: enable	
	0: prohibited	

6.5.1.13 Port P0 low-level interrupt enable configuration register (**P0LIE**)

Offset address: 0x134

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						Reserved						POL	POL	POL	POL
												IE3	IE2	IE1	IE0

Bit	mark	Function description
31:4	Reserved	
3	P0LIE3 port P03 low-level interrupt enable configuration register	<p>1: enable</p> <p>0: prohibited</p>
2	P0LIE2 port P02 low-level interrupt enable configuration register	<p>1: enable</p> <p>0: prohibited</p>
1	P0LIE1 port P01 low-level interrupt enable configuration register	<p>1: enable</p> <p>0: prohibited</p>
0	P0LIE0 port P00 low-level interrupt enable configuration register	<p>1: enable</p> <p>0: prohibited</p>

Page 118

6.5.1.14 Port P0 rising edge interrupt enable configuration register (**P0RIE**)

Offset address: 0x138

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15 14 13 12 11 10

9

5

三

Reserved

P0

P

F

IE3 IE2 IE1 IE0

RW RW RW RW

Bit	mark	Function description
31:4	Reserved	
3	P0RIE3 port P03 rising edge interrupt enable configuration register 1: enable	

0: prohibited

2 P0RIE2 port P02 rising edge interrupt enable configuration register

1: enable

0: prohibited

1 P0RIE1 port P01 rising edge interrupt enable configuration register

1: enable

0: prohibited

0 P0RIE0 port P00 rising edge interrupt enable configuration register

1: enable

0: prohibited

Page 119

6.5.1.15 Port P0 Falling Edge Interrupt Enable Configuration Register (P0FIE)

Offset address: 0x13C

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

											Reserved									
											9	8	7	6	5	4	3	2	1	0
											Reserved				P0F	P0F	P0F	P0F		
															IE3	IE2	IE1	IE0		
															RW	RW	RW	RW		

Bit mark Function description

31:4 Reserved

3 P0FIE3 Port P03 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

2 P0FIE2 Port P02 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

1 P0FIE1 Port P01 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

0 P0FIE0 Port P00 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

Page 120

6.5.1.16 Port P0 Interrupt Status Register (P0_STAT)

Offset address: 0x200

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved

P0S P0S P0S P0S

TA3 TA2 TA1 TA0

RO RO RO RO

Bit	mark	Function description
-----	------	----------------------

31:4 Reserved

3 P0STA3 Port P03 Interrupt Status Register

1: Interrupt trigger

0: No interrupt trigger

2 P0STA2 port P02 interrupt status register

1: Interrupt trigger

0: No interrupt trigger

1 P0STA1 port P01 interrupt status register

1: Interrupt trigger

0: No interrupt trigger

0 P0STA0 port P00 interrupt status register

1: Interrupt trigger

0: No interrupt trigger

Page 121**6.5.1.17 Port P0 Interrupt Clear Register (P0_ICLR)**

Offset address: 0x210

Reset value: 0xffff ffff

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

										Reserved									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	P0C	P0C	P0C	P0C
						Reserved									LR3	LR2	LR1	LR0	
															RW	RW	RW	RW	

Bit mark Function description

31:4 Reserved

3 P0CLR3 Port P03 Interrupt Clear Register

1: Reserve the interrupt flag bit

0: Clear the interrupt flag bit

2 P0CLR2 Port P02 Interrupt Clear Register

1: Reserve the interrupt flag bit

0: Clear the interrupt flag bit

1 P0CLR1 Port P01 Interrupt Clear Register

1: Reserve the interrupt flag bit

0: Clear the interrupt flag bit

0 P0CLR0 Port P00 Interrupt Clear Register

1: Reserve the interrupt flag bit

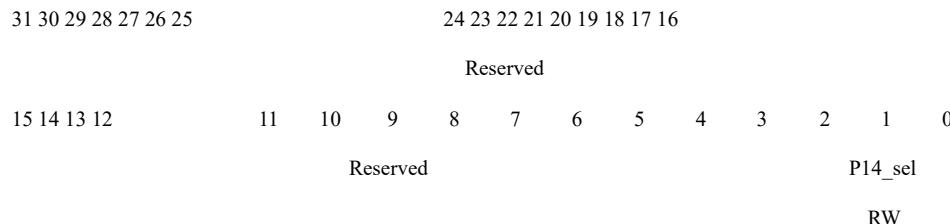
0: Clear the interrupt flag bit

6.5.2 Port P1

6.5.2.1 Port P14 Function Configuration Register (P14_SEL)

Offset address: 0x50

Reset value: 0x0000 0000



Bit mark Function description

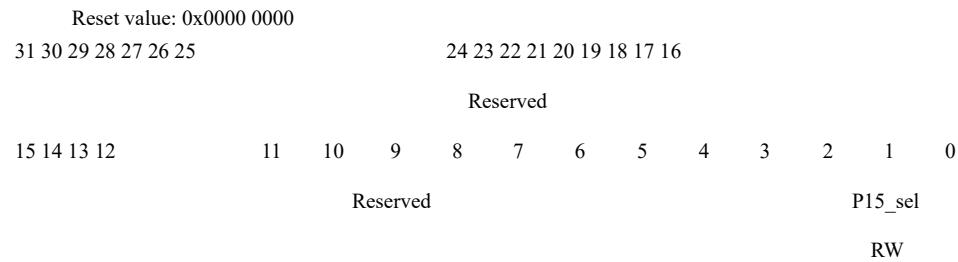
31:3 Reserved

2:0 P14_sel Port P14 function selection.

000: GPIO_P14	
001: I2C_SCL	I2C module clock signal
010: TIM2_TOGN	The reverse signal of the Timer2 module flip signal
011: PCA_ECI	PCA module external clock input signal
100: ADC_RDY	ADC module RDY signal
101: SPI_CS	SPI module host mode chip select signal
110: UART0_TXD	TXD signal of UART0 module
111: NC	

6.5.2.2 Port P15 Function Configuration Register (P15_SEL)

Offset address: 0x54



Bit	mark	Function description
-----	------	----------------------

31:3	Reserved	
------	----------	--

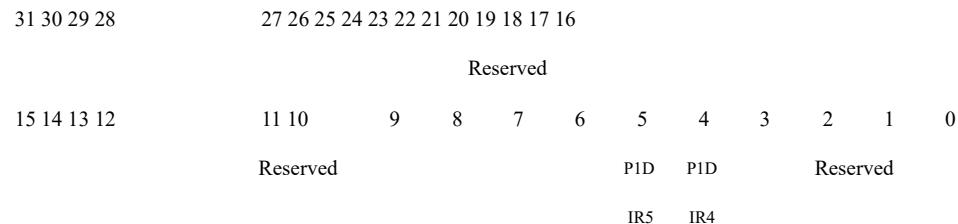
2:0	P15_sel Port P15 function selection.	
-----	--------------------------------------	--

000: GPIO_P15	
001: I2C_SDA	I2C module data signal
010: TIM2_TOG	Timer2 module flip signal
011: TIM4_CHB	Advanced Timer module channel 0 B signal
100: LPTIM_GATE	Timer3 module gate control signal
101: SPI_SCK	SPI module clock signal
110: UART0_RXD	RXD signal of UART0 module
111: LVD_OUT	LVD module output signal

6.5.2.3 Port P1 Input and Output Configuration Register (P1DIR)

Offset address: 0x140

Reset value: 0xffff ffff



Bit	mark	Function description
31:6	Reserved	
5	P1DIR5 Port P15 Input and Output Configuration Register	
	1: Configure as input	
	0: Configure as output	
4	P1DIR4 Port P14 Input and Output Configuration Register	
	1: Configure as input	
	0: Configure as output	
3:0	Reserved	

Page 125**6.5.2.4 Port P1 Input Value Register (P1IN)**

Offset address: 0x144

Reset value: NA

31 30 29 28	27 26 25 24 23 22 21 20 19 18 17 16	Reserved
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	
Reserved	P1I P1I	Reserved
	N5 N4	
	RO RO	

Bit	mark	Function description
31:6	Reserved	
5	P1IN5 Port P15 input value register	
	1: Input is high level	

		0: Input is low level
4	P1IN4	Port P14 input value register
		1: Input is high level
		0: Input is low level

HC32L110 Series User Manual Rev2.31

Page 125 of 527

Page 126

6.5.2.5 Port P1 output value configuration register (P1OUT)

Offset address: 0x148

Reset value: NA

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						Reserved			P1O	P1O			Reserved		
									UT5	UT4					
									RW	RW					

Bit mark Function description

31:6 Reserved

5 P1OUT5 Port P15 output value configuration register

1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.

0: Output low level.

4 P1OUT4 Port P14 output value configuration register

1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.

0: Output low level.

3:0 Reserved

Page 127**6.5.2.6 Port P1 Digital-to-Analog Configuration Register (P1ADS)**

Offset address: 0x14C

Reset value: 0x0000 0000

31 30 29 28	27 26 25 24 23 22 21 20 19 18 17 16	Reserved												
15 14 13 12	11 10	9	8	7	6	5	4	3	2	1	0	P1A	P1A	Reserved
	Reserved					DS5	DS4							
								RW	RW					

Bit mark Function description

31:6 Reserved

5 P1ADS5 port P15 digital-to-analog configuration register

1: Configure as an analog port

0: Configure as a digital port

4 P1ADS4 port P14 digital-to-analog configuration register

1: Configure as an analog port

0: Configure as a digital port

3:0 Reserved

Page 128

6.5.2.7 Port P1 Drive Capability Configuration Register (**P1DR**)

Offset address: 0x15C

Reset value: 0x0000 0000

27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15 14 13 12

11 10

9

6

3

1

0

Reserved

P1D P1D

Reserved

R5 R4

RW RW

31:6	Reserved	
5	P1DR5	Port P15 drive capability configuration register
		1: Low drive capability
		0: High drive capability
4	P1DR4	Port P14 drive capability configuration register
		1: Low drive capability
		0: High drive capability
3:0	Reserved	

6.5.2.8 Port P1 pull-up enable configuration register (P1PU)

Offset address: 0x160

Reset value: 0x0000 0000

Bit	mark	Function description
31:6	Reserved	
5	P1PU5	Port P15 pull-up enable configuration register
		1: enable
		0: prohibited
4	P1PU4	Port P14 pull-up enable configuration register
		1: enable
		0: prohibited
3:0	Reserved	

6.5.2.9 Port P1 pull-down enable configuration register (P1PD)

Offset address: 0x164

Reset value: 0x0000 0000

31 30 29 28	27 26 25 24 23 22 21 20 19 18 17 16	
Reserved		
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	
	Reserved	P1P P1P Reserved
		D5 D4
		RW RW

Bit	mark	Function description
31:6	Reserved	
5	P1PD5	Port P15 pull-down enable configuration register 1: enable 0: prohibited
4	P1PD4	Port P14 pull-down enable configuration register 1: enable 0: prohibited
3:0	Reserved	

6.5.2.10 Port P1 Open Drain Output Configuration Register (P1OD)

Offset address: 0x16C

Reset value: 0x0000 0000

31 30 29 28	27 26 25 24 23 22 21 20 19 18 17 16	
Reserved		
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	
	Reserved	P1O P1O Reserved
		D5 D4

RW RW

Bit	mark	Function description
31:6	Reserved	
5	P1OD5	Port P15 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
4	P1OD4	Port P14 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
3:0	Reserved	

Page 132**6.5.2.11 Port P1 high level interrupt enable configuration register (P1HIE)**

Offset address: 0x170

Reset value: 0x0000 0000

31 30 29 28	27 26 25 24 23 22 21 20 19 18 17 16	Reserved	
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	P1H	P1H
Reserved		IE5	IE4
		RW	RW

Bit	mark	Function description
31:6	Reserved	
5	P1HIE5	Port P15 High Level Interrupt Enable Configuration Register

1: enable

0: prohibited

4 P1HIE4 port P14 high level interrupt enable configuration register

1: enable

0: prohibited

3:0 Reserved

Page 133**6.5.2.12 Port P1 Low Level Interrupt Enable Configuration Register (P1LIE)**

Offset address: 0x174

Reset value: 0x0000 0000

31 30 29 28	27 26 25 24 23 22 21 20 19 18 17 16	Reserved
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	
Reserved	P1L IE5	Reserved
	IE4	RW

Bit mark Function description

31:6 Reserved

5 P1LIE5 port P15 low-level interrupt enable configuration register

1: enable

0: prohibited

4 P1LIE4 port P14 low-level interrupt enable configuration register

1: enable

0: prohibited

3:0 Reserved

HC32L110 Series User Manual Rev2.31

Page 133 of 527

Page 134

6.5.2.13 Port P1 rising edge interrupt enable configuration register (**P1RIE**)

Offset address: 0x178

Reset value: 0x0000 0000

Reserved

15 14 13 12

11 10

9

5

3

1

IE5 IE4

RW RW

RW RW

Bit mark Function description

31:6 Reserved

5 P1RIE5 port P15 rising edge interrupt enable configuration register

1; enable

0: prohibited

4 P1RJE4 port P14 rising edge interrupt enable configuration register

1: enable

0: prohibited

3:0 Reserved

Page 135

6.5.2.14 Port P1 Falling Edge Interrupt Enable Configuration Register (**P1FIE**)

Offset address: 0x17C

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Reserved

P1F P1F

Reserved

IE5 IE4

RW RW

Bit mark Function description

31:6 Reserved

5 P1FIE5 Port P15 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

4 P1FIE4 Port P14 Falling Edge Interrupt Enable Configuration Register

1: enable

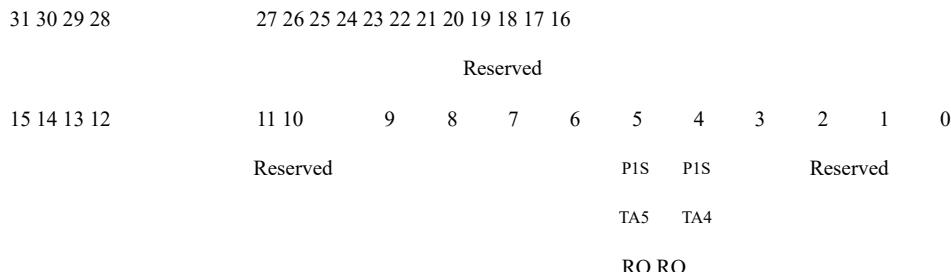
0: prohibited

3:0 Reserved

6.5.2.15 Port P1 Interrupt Status Register (P1_STAT)

Offset address: 0x240

Reset value: 0x0000 0000



Bit	mark	Function description
-----	------	----------------------

31:6 Reserved

5 P1STA5 port P15 interrupt status register

1: Interrupt trigger

0: No interrupt trigger

4 P1STA4 port P14 interrupt status register

1: Interrupt trigger

0: No interrupt trigger

3:0 Reserved

6.5.2.16 Port P1 Interrupt Clear Register (P1_ICLR)

Offset address: 0x250

Reset value: 0xffff ffff

31 30 29 28	27 26 25 24 23 22 21 20 19 18 17 16	
Reserved		
15 14 13 12	11 10	9 8 7 6 5 4 3 2 1 0
	Reserved	P1C P1C Reserved
		LR5 LR4
		RW RW

Bit	mark	Function description
-----	------	----------------------

31:6	Reserved
------	----------

5	P1CLR5 Port P15 Interrupt Clear Register
---	--

	1: Reserve the interrupt flag bit
--	-----------------------------------

	0: Clear the interrupt flag bit
--	---------------------------------

4	P1CLR4 Port P14 Interrupt Clear Register
---	--

	1: Reserve the interrupt flag bit
--	-----------------------------------

	0: Clear the interrupt flag bit
--	---------------------------------

3:0	Reserved
-----	----------

6.5.3 Port P2

6.5.3.1 Port P23 Function Configuration Register (P23_SEL)

Offset address: 0x8C

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16	
Reserved		
15 14 13 12	11 10	9 8 7 6 5 4 3 2 1 0

Reserved	P23_sel
	RW

Bit	mark	Function description
31:3	Reserved	
2:0	P23_sel	Port P23 function selection.
		000: GPIO P23
	001:	TIM6_CHA Advanced Timer module channel 2 A signal
	010:	TIM4_CHB Advanced Timer module channel 0 B signal
	011:	TIM4_CHA Advanced Timer module channel 0 A signal
	100:	PCA_CH0 PCA module channel 0 capture/compare signal
	101:	SPI_MOSI SPI module host input and slave output data signal
	110:	UART1_TXD TXD signal of UART1 module
	111:	IR_OUT Infrared output signal

6.5.3.2 Port P24 Function Configuration Register (P24_SEL)

Offset address: 0x90

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16	
	Reserved	
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	
	Reserved	P24_sel
		RW

Bit	mark	Function description
31:3	Reserved	
2:0	P24_sel	Port P24 function selection.
		000: GPIO P24

001: TIM4_CHB	Advanced Timer module channel 0 B signal
010: TIM5_CHB	Advanced Timer module channel 1 B signal
011: HCLK_OUT	AHB bus clock output signal
100: PCA_CH1	PCA module channel 1 capture/compare signal
101: SPI_MOSI	SPI module master output slave input data signal
110: UART1_RXD	RXD signal of UART1 module
111: VC1_OUT	VC1 module output

6.5.3.3 Port P25 Function Configuration Register (P25_SEL)

Offset address: 0x94

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16
	Reserved
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0
	Reserved
	P25_sel
	RW

Bit mark Function description

31:3 Reserved

2:0 P25_sel Port P25 function selection.

000: GPIO P25	
001: SPI_SCK	SPI module clock signal
010: PCA_CH0	PCA module channel 0 capture/compare signal
011: TIM5_CHA	Advanced Timer module channel 1 A signal
100: LVD_OUT	LVD module output signal
101: LPUART_RXD	LPUART module RXD signal

110: I2C_SDA	I2C module data signal
111: TIM1_GATE	Timer1 module gate control signal

Page 141**6.5.3.4 Port P26 Function Configuration Register (P26_SEL)**

Offset address: 0x98

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16	Reserved	P26_sel	RW
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	Reserved		

Bit mark Function description

31:3 Reserved

2:0 P26_sel Port P26 function selection.

000: GPIO_P26	
001: SPI_MOSI	SPI module master output slave input data signal
010: TIM4_CHA	Advanced Timer module channel 0 A signal
011: TIM5_CHB	Advanced Timer module channel 1 B signal
100: PCA_CH2	PCA module channel 2 capture/compare signal
101: LPUART_RXD	TXD signal of LPUART module
110: I2C_SCL	I2C module clock signal
111: TIM1_EXT	Timer1 module external clock input signal

Page 142**6.5.3.5 Port P27 Function Configuration Register (P27_SEL)**

Offset address: 0x9C

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16	Reserved	P27_sel	RW
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	Reserved		

Bit mark Function description

31:3 Reserved

2:0 P27_sel Port P27 function selection.

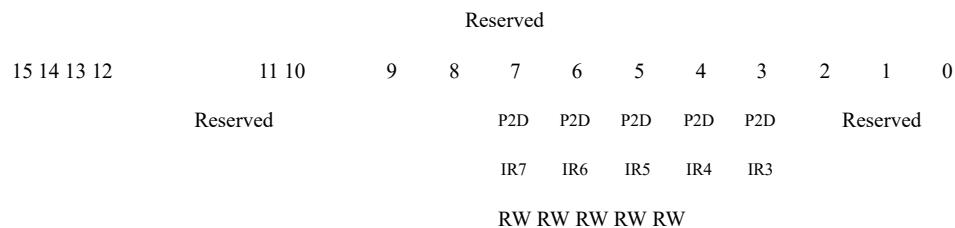
000: GPIO P27	
001: SPI_MOSI	SP I module host input and slave output data signal
010: TIM5_CHA	Advanced Timer module channel 1 A signal
011: TIM6_CHA	Advanced Timer module channel 2 A signal
100: PCA_CH3	PCA module channel 3 capture/compare signal
101: UART0_RXD	RXD signal of UART0 module
110: RCH_OUT	Internal 24M RC clock output signal
111: XTH_OUT	External 32M crystal oscillator output signal

6.5.3.6 Port P2 Input and Output Configuration Register (P2DIR)

Offset address: 0x180

Reset value: 0xffff ffff

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16



Bit mark Function description

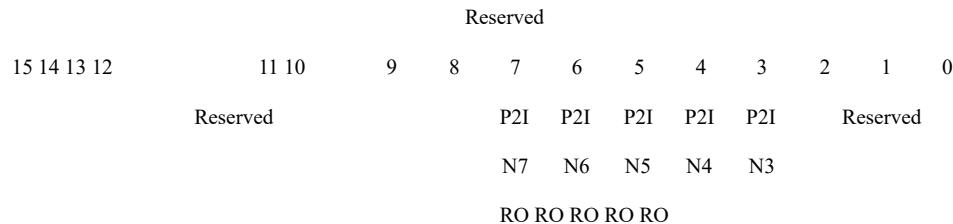
31:8	Reserved	
7	P2DIR7 Port P27 Input and Output Configuration Register	
	1: Configure as input	
	0: Configure as output	
6	P2DIR6 Port P26 Input and Output Configuration Register	
	1: Configure as input	
	0: Configure as output	
5	P2DIR5 port P25 input and output configuration register	
	1: Configure as input	
	0: Configure as output	
4	P2DIR4 port P24 input and output configuration register	
	1: Configure as input	
	0: Configure as output	
3	P2DIR3 port P23 input and output configuration register	
	1: Configure as input	
	0: Configure as output	
2:0	Reserved	

6.5.3.7 Port P2 Input Value Register (P2IN)

Offset address: 0x184

Reset value: NA

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16



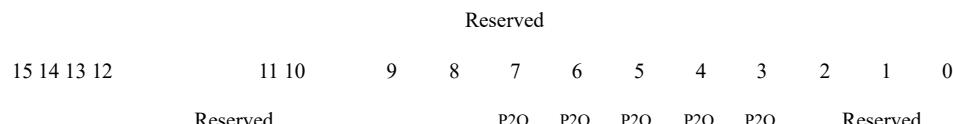
Bit	mark	Function description
31:8	Reserved	
7	P2IN7	Port P27 input value register 1: Input is high level 0: Input is low level
6	P2IN6	Port P26 input value register 1: Input is high level 0: Input is low level
5	P2IN5	Port P25 input value register 1: Input is high level 0: Input is low level
4	P2IN4	Port P24 input value register 1: Input is high level 0: Input is low level
3	P2IN3	Port P23 input value register 1: Input is high level 0: Input is low level
2:0	Reserved	

6.5.3.8 Port P2 output value configuration register (P2OUT)

Offset address: 0x188

Reset value: NA

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16



UT7 UT6 UT5 UT4 UT3

RW RW RW RW RW

Bit mark Function description

31:8 Reserved

7 P2OUT7 Port P27 output value configuration register

1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.

0: Output low level.

6 P2OUT6 Port P26 output value configuration register

1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.

0: Output low level.

5 P2OUT5 port P25 output value configuration register

1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.

0: Output low level.

4 P2OUT4 port P24 output value configuration register

1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.

0: Output low level.

3 P2OUT3 port P23 output value configuration register

1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.

0: Output low level.

2:0 Reserved

6.5.3.9 Port P2 Digital-to-Analog Configuration Register (P2ADS)

Offset address: 0x18C

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15 14 13 12

11 10

9

8

7

6

5

4

3

2

1

0

Reserved

DS7 DS6 DS5 DS4 DS3

RW RW RW RW RW

Bit mark Function description

31:8 Reserved

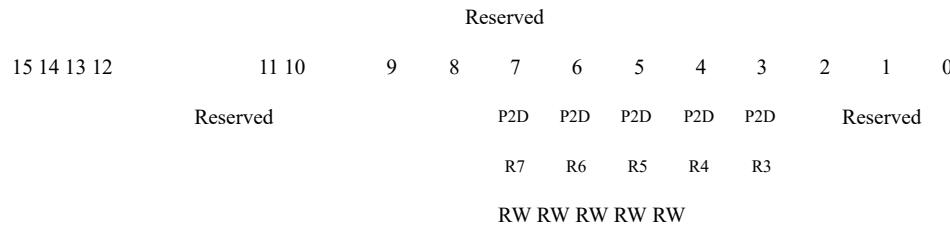
- 7 P2ADS7 port P27 digital-to-analog configuration register
 - 1: Configure as an analog port
 - 0: Configure as a digital port
- 6 P2ADS6 Port P26 Digital-to-Analog Configuration Register
 - 1: Configure as an analog port
 - 0: Configure as a digital port
- 5 P2ADS5 port P25 digital-to-analog configuration register
 - 1: Configure as an analog port
 - 0: Configure as a digital port
- 4 P2ADS4 port P24 digital-analog configuration register
 - 1: Configure as an analog port
 - 0: Configure as a digital port
- 3 P2ADS3 port P23 digital-analog configuration register
 - 1: Configure as an analog port
 - 0: Configure as a digital port
- 2:0 Reserved

Page 147**6.5.3.10 Port P2 Drive Capability Configuration Register (P2DR)**

Offset address: 0x19C

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16



- | Bit | mark | Function description |
|------|----------|--|
| 31:8 | Reserved | |
| 7 | P2DR7 | Port P27 drive capability configuration register <ul style="list-style-type: none"> 1: Low drive capability 0: High drive capability |
| 6 | P2DR6 | Port P26 drive capability configuration register <ul style="list-style-type: none"> 1: Low drive capability 0: High drive capability |

5	P2DR5	Port P25 drive capability configuration register
		1: Low drive capability
		0: High drive capability
4	P2DR4	Port P24 drive capability configuration register
		1: Low drive capability
		0: High drive capability
3	P2DR3	Port P23 drive capability configuration register
		1: Low drive capability
		0: High drive capability
2:0	Reserved	

Page 148**6.5.3.11 Port P2 pull-up enable configuration register (P2PU)**

Offset address: 0x1A0

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved				P2P	P2P	P2P	P2P	P2P		Reserved	
								U7	U6	U5	U4	U3			
								RW	RW	RW	RW	RW			

Bit	mark	Function description
31:8	Reserved	
7	P2PU7	Port P27 pull-up enable configuration register
		1: enable
		0: prohibited
6	P2PU6	Port P26 pull-up enable configuration register
		1: enable
		0: prohibited
5	P2PU5	Port P25 pull-up enable configuration register
		1: enable
		0: prohibited
4	P2PU4	Port P24 pull-up enable configuration register
		1: enable
		0: prohibited

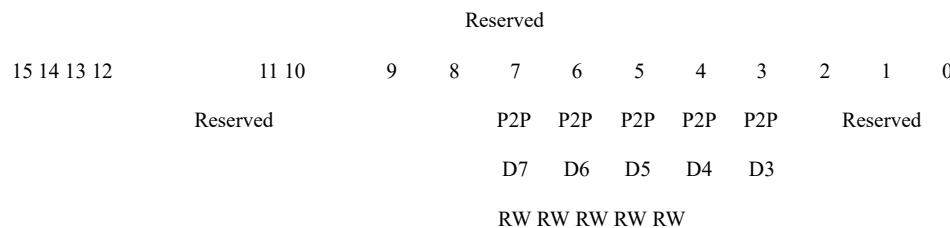
3	P2PU3	Port P23 pull-up enable configuration register
		1: enable
		0: prohibited
2:0	Reserved	

Page 149**6.5.3.12 Port P2 pull-down enable configuration register (P2PD)**

Offset address: 0x1A4

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16



Bit	mark	Function description
31:8	Reserved	
7	P2PD7	Port P27 pull-down enable configuration register
		1: enable
		0: prohibited
6	P2PD6	Port P26 pull-down enable configuration register
		1: enable
		0: prohibited
5	P2PD5	Port P25 pull-down enable configuration register
		1: enable
		0: prohibited
4	P2PD4	Port P24 pull-down enable configuration register
		1: enable
		0: prohibited
3	P2PD3	Port P23 pull-down enable configuration register
		1: enable
		0: prohibited
2:0	Reserved	

6.5.3.13 Port P2 Open Drain Output Configuration Register (P2OD)

Offset address: 0x1AC

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved												Reserved			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved		P2O	P2O	P2O	P2O	P2O	P2O				
						D7	D6	D5	D4	D3					
												RW	RW	RW	RW

Bit	mark	Function description
31:8	Reserved	
7	P2OD7	Port P27 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
6	P2OD6	Port P26 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
5	P2OD5	Port P25 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
4	P2OD4	Port P24 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
3	P2OD3	Port P23 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
2:0	Reserved	

6.5.3.14 Port P2 high level interrupt enable configuration register (P2HIE)

Offset address: 0x1B0

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved		P2H	P2H	P2H	P2H	P2H		Reserved			
						IE7	IE6	IE5	IE4	IE3					
						RW	RW	RW	RW	RW					

Bit mark Function description

31:8 Reserved

7 P2HIE7 port P27 high level interrupt enable configuration register

1: enable

0: prohibited

6 P2HIE6 Port P26 High Level Interrupt Enable Configuration Register

1: enable

0: prohibited

5 P2HIE5 port P25 high level interrupt enable configuration register

1: enable

0: prohibited

4 P2HIE4 port P24 high level interrupt enable configuration register

1: enable

0: prohibited

3 P2HIE3 port P23 high level interrupt enable configuration register

1: enable

0: prohibited

2:0 Reserved

6.5.3.15 Port P2 Low Level Interrupt Enable Configuration Register (P2LIE**)**

Offset address: 0x1B4

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	P2L IE7	P2L IE6	P2L IE5	P2L IE4	P2L IE3	Reserved
	RW	RW	RW	RW	RW	

Bit	mark	Function description
-----	------	----------------------

31:8 Reserved

7 P2LIE7 port P27 low-level interrupt enable configuration register

1: enable

0: prohibited

6 P2LIE6 Port P26 low-level interrupt enable configuration register

1: enable

0: prohibited

5 P2LIE5 port P25 low-level interrupt enable configuration register

1: enable

0: prohibited

4 P2LIE4 port P24 low-level interrupt enable configuration register

1: enable

0: prohibited

3 P2LIE3 port P23 low-level interrupt enable configuration register

1: enable

0: prohibited

2:0 Reserved

Page 153**6.5.3.16 Port P2 rising edge interrupt enable configuration register (P2RIE)**

Offset address: 0x1B8

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved	15 14 13 12	11 10	9	8	7	6	5	4	3	2	1	0
----------	-------------	-------	---	---	---	---	---	---	---	---	---	---

Reserved	P2R IE7	P2R IE6	P2R IE5	P2R IE4	P2R IE3	Reserved
	RW	RW	RW	RW	RW	

Bit	mark	Function description
-----	------	----------------------

31:8 Reserved

7 P2RIE7 port P27 rising edge interrupt enable configuration register

1: enable

0: prohibited

6 P2RIE6 Port P26 rising edge interrupt enable configuration register

1: enable

0: prohibited

5 P2RIE5 port P25 rising edge interrupt enable configuration register

1: enable

0: prohibited

4 P2RIE4 port P24 rising edge interrupt enable configuration register

1: enable

0: prohibited

3 P2RIE3 port P23 rising edge interrupt enable configuration register

1: enable

0: prohibited

2:0 Reserved

Page 154

6.5.3.17 Port P2 falling edge interrupt enable configuration register (P2FIE)

Offset address: 0x1BC

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved												Reserved			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									P2F	P2F	P2F	P2F	P2F		
									IE7	IE6	IE5	IE4	IE3		
									RW	RW	RW	RW	RW		

Bit mark Function description

31:8 Reserved

7 P2FIE7 Port P27 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

6 P2FIE6 Port P26 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

5 P2FIE5 Port P25 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

4 P2FIE4 Port P24 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

3 P2FIE3 Port P23 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

2:0 Reserved

Page 155

6.5.3.18 Port P2 Interrupt Status Register (P2_STAT)

Offset address: 0x280

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15 14 13 12	11 10	9	8	7	6	5	4	3	2	1	0
-------------	-------	---	---	---	---	---	---	---	---	---	---

Reserved	P2S	P2S	P2S	P2S	P2S	Reserved
----------	-----	-----	-----	-----	-----	----------

TA7	TA6	TA5	TA4	TA3
-----	-----	-----	-----	-----

RO RO RO RO RO

Bit	mark	Function description
-----	------	----------------------

31:8 Reserved

7 P2STA7 port P27 interrupt status register

1: Interrupt trigger

0: No interrupt trigger

6 P2STA6 port P26 interrupt status register

1: Interrupt trigger

0: No interrupt trigger

5 P2STA5 port P25 interrupt status register

1: Interrupt trigger

0: No interrupt trigger

4 P2STA4 port P24 interrupt status register

1: Interrupt trigger

	0: No interrupt trigger
3	P2STA3 port P23 interrupt status register
	1: Interrupt trigger
	0: No interrupt trigger

2:0 Reserved

Page 156**6.5.3.19 Port P2 Interrupt Clear Register (P2_ICLR)**

Offset address: 0x290

Reset value: 0xffff ffff

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved				P2C	P2C	P2C	P2C	P2C		Reserved	
								LR7	LR6	LR5	LR4	LR3			
								RW	RW	RW	RW	RW			

Bit	mark	Function description
-----	------	----------------------

31:8 Reserved

7 P2CLR7 Port P27 Interrupt Clear Register

1: Reserve the interrupt flag bit

0: Clear the interrupt flag bit

6 P2CLR6 Port P26 Interrupt Clear Register

1: Reserve the interrupt flag bit

0: Clear the interrupt flag bit

5 P2CLR5 port P25 interrupt clear register

1: Reserve the interrupt flag bit

0: Clear the interrupt flag bit

4 P2CLR4 port P24 interrupt clear register

1: Reserve the interrupt flag bit

0: Clear the interrupt flag bit

3 P2CLR3 Port P23 Interrupt Clear Register

1: Reserve the interrupt flag bit

0: Clear the interrupt flag bit

2:0 Reserved

Page 157**6.5.4 Port P3****6.5.4.1 Port P31 Function Configuration Register (P31_SEL)**

Offset address: 0xC4

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16							
Reserved								
15 14 13 12	11	10	9	8	7	6	5	4
Reserved							P31_sel	
							RW	

Bit	mark	Function description
-----	------	----------------------

31:3 Reserved

2:0 P31_sel Port P31 function selection.

000: GPIO P31	
001: LPTIM_TOG	Timer3 module flip signal
010: PCA_ECI	PCA module external clock input signal
011: PCLK_OUT	APB bus clock output signal
100: VC0_OUT	VC0 module output
101: UART0_RXD	TXD signal of UART0 module
110: RCL_OUT	Internal 38K RC clock output signal
111: HCLK_OUT	AHB bus clock output signal

Page 158**6.5.4.2 Port P32 Function Configuration Register (P32_SEL)**

Offset address: 0xC8

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16							
Reserved								
15 14 13 12	11	10	9	8	7	6	5	4
Reserved			P32_sel			RW		

Bit mark Function description

31:3 Reserved

2:0 P32_sel Port P32 function selection.

000: GPIO P32	
001: LPTIM_TOGN	Reverse signal of Timer3 module flip signal
010: PCA_CH2	PCA module channel 2 capture/compare signal
011: TIM6_CHB	Advanced Timer module channel 2 B signal
100: VC1_OUT	VC1 module output
101: UART1_TXD	TXD signal of UART1 module
110: PCA_CH4	PCA module channel 4 capture/compare signal
111: RTC_1Hz	RTC module 1Hz output signal

6.5.4.3 Port P33 Function Configuration Register (P33_SEL)

Offset address: 0xCC

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16							
Reserved								
15 14 13 12	11	10	9	8	7	6	5	4
Reserved			P33_sel			RW		

Reserved

P33_sel

RW

Bit	mark	Function description
31:3	Reserved	
2:0	P33_sel	Port P33 function selection.
	000: GPIO_P33	
	001: LPUART_RXD	LPUART module RXD signal
	010: PCA_CH1	PCA module channel 1 capture/compare signal
	011: TIM5_CHB	Advanced Timer module channel 1 B signal
	100: PCA_ECI	PCA module external clock input signal
	101: UART1_RXD	RXD signal of UART1 module
	110: XTL_OUT	External 32K crystal oscillator output signal
	111: TIM1_TOGN	The reverse signal of the Timer1 module flip signal

6.5.4.4 Port P34 Function Configuration Register (P34_SEL)

Offset address: 0xD0

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16	Reserved	
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0	Reserved	P34_sel
			RW

Bit	mark	Function description
31:3	Reserved	
2:0	P34_sel	Port P34 function selection.

000: GPIO P34	
001: PCA_CH0	PCA module channel 0 capture/compare signal
010: LPUART_TXD	TXD signal of LPUART module
011: TIM5_CHA	Advanced Timer module channel 1 A signal
100: TIM0_EXT	Timer0 module external clock input signal
101: TIM4_CHA	Advanced Timer module channel 0 A signal
110: RTC_1Hz	RTC module 1Hz output signal
111: TIM1_TOG	Timer1 module flip signal

Page 161**6.5.4.5 Port P35 Function Configuration Register (P35_SEL)**

Offset address: 0xD4

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16
Reserved	
15 14 13 12	11 10 9 8 7 6 5 4 3 2 1 0
	P35_sel

Reserved

RW

Bit mark Function description

31:3 Reserved

2:0 P35_sel Port P35 function selection.

000: GPIO P35	
001: UART1_TXD	TXD signal of UART1 module
010: TIM6_CHB	Advanced Timer module channel 2 B signal
011: UART0_TXD	TXD signal of UART0 module
100: TIM0_GATE	Timer0 module gate control signal

101: TIM4_CHB	Advanced Timer module channel 0 B signal
110: SPI_MOSI	SPI module host input and slave output data signal
111: I2C_SDA	I2C module data signal

Page 162**6.5.4.6 Port P36 Function Configuration Register (P36_SEL)**

Offset address: 0xD8

Reset value: 0x0000 0000

31 30 29 28 27 26 25	24 23 22 21 20 19 18 17 16
15 14 13 12	Reserved
11 10 9 8 7 6 5 4 3 2 1 0	P36_sel
Reserved	RW

Bit mark Function description

31:3 Reserved

2:0 P36_sel Port P36 function selection.

000: GPIO P36	
001: UART1_RXD	RXD signal of UART1 module
010: TIM6_CHA	Advanced Timer module channel 2 A signal
011: UART0_RXD	RXD signal of UART0 module
100: PCA_CH4	PCA module channel 4 capture/compare signal
101: TIM5_CHA	Advanced Timer module channel 1 A signal
110: SPI_MOSI	SPI module master output slave input data signal
111: I2C_SCL	I2C module clock signal

Page 163**6.5.4.7 Port P3 Input and Output Configuration Register (P3DIR)**

Offset address: 0x1C0

Reset value: 0xffff ffff

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved					P3D	P3D	P3D	P3D	P3D	P3D	Res
									IR6	IR5	IR4	IR3	IR2	IR1	
									RW						

Bit mark Function description

31:7 Reserved

6 P3DIR6 Port P36 Input and Output Configuration Register

1: Configure as input

0: Configure as output

5 P3DIR5 port P35 input and output configuration register

1: Configure as input

0: Configure as output

4 P3DIR4 port P34 input and output configuration register

1: Configure as input

0: Configure as output

3 P3DIR3 port P33 input and output configuration register

1: Configure as input

0: Configure as output

2 P3DIR2 port P32 input and output configuration register

1: Configure as input

0: Configure as output

1 P3DIR1 port P31 input and output configuration register

Page 164

1: Configure as input

0: Configure as output

0 Reserved

Page 165

6.5.4.8 Port P3 Input Value Register (P3IN)

Offset address: 0x1C4

Reset value: NA

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						Reserved			P3I	P3I	P3I	P3I	P3I	P3I	Res
									N6	N5	N4	N3	N2	N1	
RO RO RO RO RO RO															

Bit	mark	Function description
31:7	Reserved	
6	P3IN6	Port P36 input value register 1: Input is high level 0: Input is low level
5	P3IN5	Port P35 input value register 1: Input is high level 0: Input is low level
4	P3IN4	Port P34 input value register 1: Input is high level 0: Input is low level
3	P3IN3	Port P33 input value register 1: Input is high level 0: Input is low level
2	P3IN2	Port P32 input value register 1: Input is high level 0: Input is low level
1	P3IN1	Port P31 input value register 1: Input is high level 0: Input is low level
0	Reserved	

Page 16**6.5.4.9 Port P3 output value configuration register (**P3OUT**)**

Offset address: 0x1C8

Reset value: NA

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15 14 13 12	11 10	9	8	7	6	5	4	3	2	1	0
Reserved					P3O	P3O	P3O	P3O	P3O	P3O	Res
					UT6	UT5	UT4	UT3	UT2	UT1	

RW RW RW RW RW RW

Bit mark Function description

- 31:7 Reserved
- 6 P3OUT6 Port P36 output value configuration register
- 1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.
 - 0: Output low level.
- 5 P3OUT5 port P35 output value configuration register
- 1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.
 - 0: Output low level.
- 4 P3OUT4 Port P34 output value configuration register
- 1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.
 - 0: Output low level.
- 3 P3OUT3 Port P33 output value configuration register
- 1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.
 - 0: Output low level.
- 2 P3OUT2 port P32 output value configuration register
- 1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.
 - 0: Output low level.
- 1 P3OUT1 port P31 output value configuration register

Page 168

- 1: Output high level. If it is configured as an open-drain output, an external pull-up resistor is required to pull it high.
- 0: Output low level.
- 0 Reserved

Page 169**6.5.4.10 Port P3 Digital-to-Analog Configuration Register (P3ADS)**

Offset address: 0x1CC

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved					P3A	P3A	P3A	P3A	P3A	P3A	Res
									DS6	DS5	DS4	DS3	DS2	DS1	
									RW	RW	RW	RW	RW	RW	

Bit mark Function description

31:7 Reserved

6 P3ADS6 Port P36 Digital-to-Analog Configuration Register

1: Configure as an analog port

0: Configure as a digital port

5 P3ADS5 Port P35 Digital-to-Analog Configuration Register

1: Configure as an analog port

0: Configure as a digital port

4 P3ADS4 port P34 digital-to-analog configuration register

1: Configure as an analog port

0: Configure as a digital port

3 P3ADS3 port P33 digital-to-analog configuration register

1: Configure as an analog port

- 0: Configure as a digital port
- 2 P3ADS2 Port P32 Digital-to-Analog Configuration Register
 - 1: Configure as an analog port
 - 0: Configure as a digital port
- 1 P3ADS1 port P31 digital-analog configuration register

Page 170

- 1: Configure as an analog port
- 0: Configure as a digital port
- 0 Reserved

Page 171**6.5.4.11 Port P3 Drive Capability Configuration Register (P3DR)**

Offset address: 0x1DC

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved					P3D	P3D	P3D	P3D	P3D	P3D	Res
									R6	R5	R4	R3	R2	R1	
									RW	RW	RW	RW	RW	RW	

Bit	mark	Function description
31:7	Reserved	
6	P3DR6	Port P36 drive capability configuration register 1: Low drive capability 0: High drive capability
5	P3DR5	Port P35 drive capability configuration register 1: Low drive capability 0: High drive capability
4	P3DR4	Port P34 drive capability configuration register 1: Low drive capability 0: High drive capability
3	P3DR3	Port P33 drive capability configuration register 1: Low drive capability 0: High drive capability
2	P3DR2	Port P32 drive capability configuration register 1: Low drive capability 0: High drive capability
1	P3DR1	Port P31 drive capability configuration register

1: Low drive capability

0: High drive capability

0 Reserved

6.5.4.12 Port P3 pull-up enable configuration register (**P3PU**)

Offset address: 0x1E0

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15 14 13 12	11 10	9	8	7	6	5	4	3	2	1	0
	Reserved				P3P	P3P	P3P	P3P	P3P	P3P	Res
					U6	U5	U4	U3	U2	U1	

RW RW RW RW RW RW

Bit	mark	Function description
31:7	Reserved	
6	P3PU6	Port P36 pull-up enable configuration register 1: enable 0: prohibited
5	P3PU5	Port P35 pull-up enable configuration register 1: enable 0: prohibited
4	P3PU4	Port P34 pull-up enable configuration register 1: enable 0: prohibited
3	P3PU3	Port P33 pull-up enable configuration register 1: enable 0: prohibited
2	P3PU2	Port P32 pull-up enable configuration register 1: enable 0: prohibited
1	P3PU1	Port P31 pull-up enable configuration register 1: enable 0: prohibited
0	Reserved	

1: enable
0: prohibited

Page 175**6.5.4.13 Port P3 pull-down enable configuration register (P3PD)**

Offset address: 0x1E4

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15 14 13 12	11 10	9	8	7	6	5	4	3	2	1	0
	Reserved				P3P	P3P	P3P	P3P	P3P	P3P	Res
					D6	D5	D4	D3	D2	D1	

RW RW RW RW RW RW

Bit mark Function description

31:7 Reserved

6 P3PD6 Port P36 pull-down enable configuration register

1: enable

0: prohibited

5 P3PD5 Port P35 pull-down enable configuration register

1: enable

		0: prohibited
4	P3PD4	Port P34 pull-down enable configuration register
		1: enable
		0: prohibited
3	P3PD3	Port P33 pull-down enable configuration register
		1: enable
		0: prohibited
2	P3PD2	Port P32 pull-down enable configuration register
		1: enable
		0: prohibited
1	P3PD1	Port P31 pull-down enable configuration register

Page 176

		1: enable
		0: prohibited
0	Reserved	

Page 177**6.5.4.14 Port P3 Open Drain Output Configuration Register (P3OD)**

Offset address: 0x1EC

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved					P3O	P3O	P3O	P3O	P3O	P3O	Res
									D6	D5	D4	D3	D2	D1	
									RW	RW	RW	RW	RW	RW	

Bit	mark	Function description
31:7	Reserved	
6	P3OD6	Port P36 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
5	P3OD5	Port P35 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
4	P3OD4	Port P34 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
3	P3OD3	Port P33 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
2	P3OD2	Port P32 open-drain output configuration register 1: Set the port output mode to open-drain output 0: Set the port output mode to push-pull output
1	P3OD1	Port P31 open-drain output configuration register

Page 178

1: Set the port output mode to open-drain output

0: Set the port output mode to push-pull output

0 Reserved

Page 179

6.5.4.15 Port P3 high level interrupt enable configuration register (P3HIE)

Offset address: 0x1F0

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved					P3H	P3H	P3H	P3H	P3H	P3H	Res
									IE6	IE5	IE4	IE3	IE2	IE1	

RW RW RW RW RW RW

Bit mark Function description

31:7 Reserved

6 P3HIE6 Port P36 High Level Interrupt Enable Configuration Register

1: enable

0: prohibited

5 P3HIE5 port P35 high level interrupt enable configuration register

1: enable

0: prohibited

4 P3HIE4 port P34 high level interrupt enable configuration register

1: enable

0: prohibited

3 P3HIE3 Port P33 High Level Interrupt Enable Configuration Register

1: enable

0: prohibited

2 P3HIE2 Port P32 High Level Interrupt Enable Configuration Register

1: enable

0: prohibited

1 P3HIE1 port P31 high level interrupt enable configuration register

HC32L110 Series User Manual Rev2.31

Page 179 of 527

Page 180

1: enable

0: prohibited

0 Reserved

Page 181**6.5.4.16 Port P3 Low Level Interrupt Enable Configuration Register (P3LIE)**

Offset address: 0x1F4

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					P3L	Res									
					IE6	IE5	IE4	IE3	IE2	IE1					
					RW	RW	RW	RW	RW	RW					

Bit	mark	Function description
31:7	Reserved	
6	P3LIE6 Port P36 low-level interrupt enable configuration register	
	1: enable	
	0: prohibited	
5	P3LIE5 port P35 low-level interrupt enable configuration register	
	1: enable	
	0: prohibited	
4	P3LIE4 port P34 low-level interrupt enable configuration register	
	1: enable	
	0: prohibited	
3	P3LIE3 port P33 low-level interrupt enable configuration register	
	1: enable	
	0: prohibited	
2	P3LIE2 port P32 low-level interrupt enable configuration register	
	1: enable	
	0: prohibited	
1	P3LIE1 port P31 low-level interrupt enable configuration register	
	1: enable	
	0: prohibited	

1	1: enable
	0: prohibited
0	Reserved

Page 183**6.5.4.17 Port P3 rising edge interrupt enable configuration register (**P3RIE**)**

Offset address: 0x1F8

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										P3R	P3R	P3R	P3R	P3R	Res
										IE6	IE5	IE4	IE3	IE2	IE1
										RW	RW	RW	RW	RW	RW

Bit mark Function description

31:7 Reserved

6 P3RIE6 Port P36 rising edge interrupt enable configuration register

1: enable

0: prohibited

5 P3RIE5 port P35 rising edge interrupt enable configuration register

1: enable

0: prohibited

4 P3RIE4 port P34 rising edge interrupt enable configuration register

1: enable

0: prohibited

3 P3RIE3 port P33 rising edge interrupt enable configuration register

1: enable

0: prohibited

2 P3RIE2 port P32 rising edge interrupt enable configuration register

1: enable

0: prohibited

1 P3RIE1 port P31 rising edge interrupt enable configuration register

Page 184

1: enable

0: prohibited

0 Reserved

Page 185**6.5.4.18 Port P3 falling edge interrupt enable configuration register (P3FIE)**

Offset address: 0x1FC

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved					P3F	P3F	P3F	P3F	P3F	P3F	Res
									IE6	IE5	IE4	IE3	IE2	IE1	
									RW	RW	RW	RW	RW	RW	

Bit mark Function description

31:7 Reserved

6 P3FIE6 Port P36 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

5 P3FIE5 Port P35 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

4 P3FIE4 Port P34 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

3 P3FIE3 Port P33 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

2 P3FIE2 Port P32 Falling Edge Interrupt Enable Configuration Register

1: enable

0: prohibited

1 P3FIE1 Port P31 Falling Edge Interrupt Enable Configuration Register

Page 186

1: enable

0: prohibited

0 Reserved

6.5.4.19 Port P3 Interrupt Status Register (P3_STAT)

Offset address: 0x2C0

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved					P3S	P3S	P3S	P3S	P3S	P3S	Res
									TA6	TA5	TA4	TA3	TA2	TA1	
RO RO RO RO RO RO															

Bit	mark	Function description
-----	------	----------------------

31:7 Reserved

6 P3STA6 Port P36 Interrupt Status Register

 1: Interrupt trigger

 0: No interrupt trigger

5 P3STA5 port P35 interrupt status register

 1: Interrupt trigger

 0: No interrupt trigger

4 P3STA4 port P34 interrupt status register

 1: Interrupt trigger

 0: No interrupt trigger

3 P3STA3 port P33 interrupt status register

 1: Interrupt trigger

 0: No interrupt trigger

2 P3STA2 port P32 interrupt status register

 1: Interrupt trigger

 0: No interrupt trigger

1 P3STA1 port P31 interrupt status register

 1: Interrupt trigger

 0: No interrupt trigger

0 Reserved

Page 189**6.5.4.20 Port P3 Interrupt Clear Register (P3_ICLR)**

Offset address: 0x2D0

Reset value: 0xffff ffff

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15 14 13 12	11 10	9	8	7	6	5	4	3	2	1	0
	Reserved				P3C	P3C	P3C	P3C	P3C	P3C	Res
					LR6	LR5	LR4	LR3	LR2	LR1	

RW RW RW RW RW RW

Bit mark Function description

31:7 Reserved

6 P3CLR6 Port P36 Interrupt Clear Register

1: Reserve the interrupt flag bit

0: Clear the interrupt flag bit

5 P3CLR5 Port P35 Interrupt Clear Register

- 1: Reserve the interrupt flag bit
 - 0: Clear the interrupt flag bit
- 4 P3CLR4 Port P34 Interrupt Clear Register
- 1: Reserve the interrupt flag bit
 - 0: Clear the interrupt flag bit
- 3 P3CLR3 Port P33 Interrupt Clear Register
- 1: Reserve the interrupt flag bit
 - 0: Clear the interrupt flag bit
- 2 P3CLR2 Port P32 Interrupt Clear Register
- 1: Reserve the interrupt flag bit
 - 0: Clear the interrupt flag bit
- 1 P3CLR1 Port P31 Interrupt Clear Register

Page 190

- 1: Reserve the interrupt flag bit
 - 0: Clear the interrupt flag bit
- 0 Reserved

Page 191**6.5.5 Port Auxiliary Function****6.5.5.1 Port auxiliary function configuration register 1 (GPIO_CTRL1)**

Offset address: 0x304

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	ir_p	hclk	pclk	hclk_sel	pclk_sel				ssn_sel			ext_clk_sel			
ol		_en		_en											
RW	RW	RW		RW		RW		RW		RW		RW			

Bit mark Function description

31:15	Reserved	
14	ir_pol	IR output polarity selection. 0-positive output 1-Reverse output
13	hclk_en	hclk output control. 0-output 0 1-output hclk
12	pclk_en	pclk output gating. 0-output 0 1-output pclk
11:10	hclk_sel	hclk output frequency division selection 00: hclk 01: hclk/2 10: hclk/4

Page 192

9:8	pclk_sel	pclk output frequency division selection
	00: hcclk	
	01: hcclk/2	
	10: hcclk/4	
	11: hcclk/8	
7:4	ssn_sel	SPI SSN signal source selection
	0000: high level	1000: P23
	0001: P35	1001: P24
	0010: P36	1010: P25
	0011: P01	1011: P26
	0100: P02	1100: P27
	0101: P03	1101: P31
	0110: P15	1110: P32
	0111: P14	1111: P33
3:0	ext_clk_sel	External clock signal source selection
	0000: high level	1000: P23
	0001: P35	1001: P24
	0010: P36	1010: P25
	0011: P01	1011: P26
	0100: P02	1100: P27
	0101: P03	1101: P31
	0110: P15	1110: P32
	0111: P14	1111: P33

Page 193

6.5.5.2 Port auxiliary function configuration register 2 (GPIO_CTRL2)

Offset address: 0x308

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				pca_cap4		pca_cap3		pca_cap2		pca_cap1		pca_cap0			
				_sel		_sel		_sel		_sel		_sel			
				RW		RW		RW		RW		RW			

Bit	mark	Function description
-----	------	----------------------

31:10 Reserved

9:8 pca_cap4_sel PCA capture channel 4 signal source selection

00: PCA_CH4

01: UART0_RXD

10: UART1_RXD

11: LPUART_RXD

7:6 pca_cap3_sel PCA capture channel 3 signal source selection

00: PCA_CH3

01: UART0_RXD

10: UART1_RXD

11: LPUART_RXD

5:4 pca_cap2_sel PCA capture channel 2 signal source selection

00: PCA_CH2

01: UART0_RXD

10: UART1_RXD

11: LPUART_RXD

3:2 pca_cap1_sel PCA capture channel 1 signal source selection

00: PCA_CH1

01: UART0_RXD

10: UART1_RXD

11: LPUART_RXD

1:0 pca_cap0_sel PCA capture channel 0 signal source selection
 00: PCA_CH0
 01: UART0_RXD
 10: UART1_RXD
 11: LPUART_RXD

Page 195**6.5.5.3 Port auxiliary function configuration register 3 (GPIO_CTRL3)**

Offset address: 0x30C

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					tm6_a_sel	tm5_a_sel	tm4_a_sel	tm6_b_sel	tm5_b_sel	tm4_b_sel					
	RW				RW		RW		RW		RW				RW

Bit	mark	Function description
-----	------	----------------------

31:12 Reserved

11:10 tm6_a_sel Timer6 A channel signal source selection

- 00: TIM6_CHA
- 01: UART0_RXD
- 10: UART1_RXD
- 11: LPUART_RXD

9:8 tm5_a_sel Timer5 A channel signal source selection

- 00: TIM5_CHA
- 01: UART0_RXD
- 10: UART1_RXD
- 11: LPUART_RXD

7:6 tm4_a_sel Timer4 A channel signal source selection

- 00: TIM4_CHA
- 01: UART0_RXD
- 10: UART1_RXD
- 11: LPUART_RXD

5:4 tm6_b_sel Timer6 B channel signal source selection

- 00: TIM6_CHB

Page 196

01: UART0_RXD

10: UART1_RXD

11: LPUART_RXD

3:2 tm5_b_sel Timer5 Channel B signal source selection

- 00: TIM5_CHB
- 01: UART0_RXD
- 10: UART1_RXD
- 11: LPUART_RXD

1:0 tm4_b_sel Timer4 B channel signal source selection

- 00: TIM4_CHB
- 01: UART0_RXD
- 10: UART1_RXD
- 11: LPUART_RXD

Page 197**6.5.5.4 Port auxiliary function configuration register 4 (GPIO_CTRL4)**

Offset address: 0x310

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

Reserved

15 14 13 12 11	10 9 8 7 6 5 4 3 2 1 0	tm3_gate_	tm2_gate_	tm1_gate_	tm0_gate_	
Reserved		sel	sel	sel	sel	
		RW	RW	RW	RW	

Bit mark Function description

31:8 Reserved

7:6 tm3_gate_sel Timer3 gate control input signal source selection

00: LPTIM_GATE

01: UART0_RXD

10: UART1_RXD

11: LPUART_RXD

5:4 tm2_gate_sel Timer2 gate control input signal source selection

00: TIM2_GATE

01: UART0_RXD

10: UART1_RXD

11: LPUART_RXD

3:2 tm1_gate_sel Timer1 gate control input signal source selection

 00: TIM1_GATE
 01: UART0_RXD
 10: UART1_RXD
 11: LPUART_RXD

1:0 tm0_gate_sel Timer0 gate control input signal source selection

00: TIM0_GATE
01: UART0_RXD
10: UART1_RXD
11: LPUART_RXD

Page 199

7 FLASH controller (FLASH)

7.1 Overview

This device contains a 32kByte FLASH memory, divided into 64 sectors, each sector

The capacity is 512Byte. This module supports erasing, programming and reading operations of the memory. In addition, this module supports

Support the protection of flash memory erasing and writing, as well as the write protection of control registers.

7.2 Block Diagram

Address range	Sector number	Address range	Sector number	
0x1E00-0x1FFF	Sector15	...	0x7E00-0x7FFF	Sector63
0x1C00-0x1DFF	Sector14	...	0x7C00-0x7DFF	Sector62
0x1A00-0x1BFF	Sector13	...	0x7A00-0x7BFF	Sector61
0x1800-0x19FF	Sector12	...	0x7800-0x79FF	Sector60
0x1600-0x17FF	Sector11	...	0x7600-0x77FF	Sector59
0x1400-0x15FF	Sector10	...	0x7400-0x75FF	Sector58
0x1200-0x13FF	Sector9	...	0x7200-0x73FF	Sector57
0x1000-0x11FF	Sector8	...	0x7000-0x71FF	Sector56
0x0E00-0x0FFF	Sector7	...	0x6E00-0x6FFF	Sector55
0x0C00-0x0DFF	Sector6	...	0x6C00-0x6DFF	Sector54
0x0A00-0x0BFF	Sector5	...	0x6A00-0x6BFF	Sector53
0x0800-0x09FF	Sector4	...	0x6800-0x69FF	Sector52
0x0600-0x07FF	Sector3	...	0x6600-0x67FF	Sector51
0x0400-0x05FF	Sector2	...	0x6400-0x65FF	Sector50
0x0200-0x03FF	Sector1	...	0x6200-0x63FF	Sector49
0x0000-0x01FF	Sector0	...	0x6000-0x61FF	Sector48

Figure 7-1 Division of memory sectors

Page 200

7.3 Functional description

The controller supports three bit-width modes of FLASH reading and writing: Byte (8bits), Half-word (16bits), Word (32bits). Note that the address of Byte operation must be aligned with Byte, and the target address of Half-word operation must be aligned according to Half-word (the lowest bit of the address is 0), the address of Word operation must be aligned according to Word (address The lowest two bits are 0). If the address of the read and write operation is not aligned in the above way, the system will enter Hard Fault Interrupt when an error occurs.

7.3.1 Page Erase (Sector Erase)

Page erasing can erase one page (Sector) specified by the user at a time. After the erase operation is completed, the page (Sector) The data are all 0xFF. If the erase operation is executed from FLASH, the CPU will stop fetching instructions and the hardware will automatically Wait for the operation to complete (FLASH_CR.BUSY becomes 0); if the erase operation is executed from RAM, then The CPU will not stop fetching instructions, and the user software should wait for the completion of the operation (FLASH_CR.BUSY becomes 0).

The steps for erasing a page (Sector) are as follows:

Step1 : Configure FLASH erasing parameters, see the chapter of erasing sequence for details.

Step2 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register to enable register rewriting.

Step3 : Configure FLASH_CR.OP to 2, set the Flash operation mode to Sector erase.

Step4 : Check if FLASH_CR.OP is 2, if it is not 2, jump to Step2.

Step5 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register sequentially to enable register rewriting.

Step6 : Set the corresponding bit of FLASH_SLOCK to 1, remove the erase and write protection of the sector.

Step7 : Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, jump to Step5.

Step8 : Write any data to any address in the Sector to be erased, and trigger the Sector erase.

Example: `*((unsigned char *) 0x00000200) = 0x00 .`

Step9 : Wait for FLASH_CR.BUSY to change to 0, and the Sector erase operation is completed.

Step10 : To erase other **Sectors**, repeat Step5-Step9.

7.3.2 Chip Erase (Chip the Erase)

Full chip erase can erase all pages (Sector) at once. After the erase operation is completed, all pages (Sector) in

The data are all 0xFF. If the erase operation is executed from FLASH, the CPU will stop fetching instructions and the hardware will automatically Wait for the operation to complete (FLASH_CR.BUSY becomes 0); if the erase operation is executed from RAM, then The CPU will not stop fetching instructions, and the user software should wait for the completion of the operation (FLASH_CR.BUSY becomes 0). The operation steps of full-chip erasing are as follows:

- Step1** : Configure FLASH erasing parameters, see the chapter of erasing sequence for details.
- Step2** : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register to enable register rewriting.
- Step3** : Configure FLASH_CR. OP to 3, set the Flash operation mode to Chip erase.
- Step4** : Check if FLASH_CR. OP is 3, if not 3, jump to Step2.
- Step5** : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register sequentially to enable register rewriting.
- Step6** : Set FLASH_SLOCK to 0xFFFF, and remove the erase and write protection of all Sectors.
- Step7** : Check if FLASH_SLOCK is 0xFFFF, if it is not 0xFFFF, jump to Step5.
- Step8** : Write any address in the Chip to be erased to trigger Chip erase.
- Example: `*((unsigned char *) 0x00000000) = 0x00.`
- Step9** : Wait for FLASH_CR. BUSY to become 0, and Chip erase operation is completed.

7.3.3 Write operation (Program)

Write operation can only write the bit data in FLASH from 1 to 0, so before writing data, make sure that the address to be written is The data is 0xFF. Supports writing three data lengths: Byte (8bits), Half-word (16bits), Word (32bits), The written data is stored in FLASH in little-endian mode, that is, the low address stores the low byte of the data. If the write operation is Execute from FLASH, the CPU will stop fetching instructions, and the hardware will automatically wait for the operation to complete (FLASH_CR. BU Becomes 0); if the write operation is executed from the RAM, the CPU will not stop fetching instructions, and the user software should wait for the The operation is complete (FLASH_CR. BUSY becomes 0).

Byte write operation steps are as follows:

- Step1** : Configure FLASH erasing parameters, see the chapter of erasing sequence for details.
- Step2** : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register to enable register rewriting.
- Step3** : Configure FLASH_CR. OP to 1, set the Flash operation mode to write.
- Step4** : Check if FLASH_CR. OP is 1, if it is not 1, jump to Step2.
- Step5** : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register sequentially to enable register rewriting.

- Step6** : Set the corresponding bit of FLASH_SLOCK to 1, remove the erase and write protection.
- Step7** : Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, jump to Step5.
- Step8** : Perform a Byte write operation on the target address to be written to trigger the write operation.
- Example: `*((unsigned char *) 0x00001231) = 0x5A.`
- Step9** : Wait for FLASH_CR. BUSY to become 0, and the write operation is completed.
- Step10** : If you need to write Byte to other addresses whose erasing protection has been removed, repeat Step8-Step9.

Half-word write operation steps are as follows:

- Step1** : Configure the FLASH erasing time, see the erasing sequence chapter for details.
- Step2** : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register to enable register rewriting.

Step3 : Configure FLASH_CR. OP to 1, set the Flash operation mode to write.

Step4 : Check if FLASH_CR. OP is 1, if it is not 1, jump to Step2.

Step5 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register sequentially to enable register rewriting.

Step6 : Set the corresponding bit of FLASH_SLOCK to 1, remove the erase and write protection.

Step7 : Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, then jump to Step5

Step8 : Perform a Half-word write operation on the target address to be written to trigger the write operation.

Example: `*((unsigned short *) 0x00001232) = 0xABCD`.

Step9 : Wait for FLASH_CR. BUSY to become 0, and the write operation is completed.

Step10 : If you need to write Half-word to other addresses whose erasing protection has been removed, repeat Step8-Step9.

Word writing operation steps are as follows:

Step1 : Configure FLASH erasing parameters, see the chapter of erasing sequence for details.

Step2 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register to enable register rewriting.

Step3 : Configure FLASH_CR. OP to 1, set the Flash operation mode to write.

Step4 : Check if FLASH_CR. OP is 1, if it is not 1, jump to Step2.

Step5 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register sequentially to enable register rewriting.

Step6 : Set the corresponding bit of FLASH_SLOCK to 1, remove the erase and write protection.

Step7 : Check whether the corresponding bit of FLASH_SLOCK is 1, if it is not 1, then jump to Step5

Step8 : Perform Word write operation on the target address to be written to trigger the write operation.

Example: `*((unsigned long *) 0x00001234) = 0x55667788`.

Step9 : Wait for FLASH_CR. BUSY to become 0, and the write operation is completed.

Step10 : If you need to write Word to other addresses whose erasing protection has been removed, repeat Step8-Step9.

7.3.4 Read operation

Support read 3 kinds of data length: Byte (8bits), Half-word (16bits), Word (32bits), read data

It is little-endian mode, that is, the low address stores the low byte of the data. No steps are required for reading operation, and it can be read at any time

Data in FLASH.

Byte read operation example: `temp = * ((unsigned char *) 0x00001231)`

Half-word read operation example `temp = * ((unsigned int *) 0x00001232)`

Word read operation example `temp = * ((unsigned long *) 0x00001234)`

Page 204

7.4 Erase and write timing

FLASH memory has strict time requirements for the control signals of erasing and programming operations, and the timing of the control signals is not Qualification will cause the erase operation and programming operation to fail. The erase and write parameters when HCLK is 4MHz are loaded by default. If the HCLK frequency when erasing and writing the Flash is not 4MHz, the user program should load the HCLK frequency corresponding erasing and writing parameters. When operating FLASH, the frequency range of HCLK is required to be 1MHz~32MHz. The registers related to the erase and write timing parameters are: FLASH_TNVS, FLASH_TPGS, FLASH_TPROG, FLASH_TSERASE, FLASH_TMERASE, FLASH_TPRCV, FLASH_TSRCV, FLASH_TMRCV. If HCLK is increased from the default 4MHz to 8MHz, the above FLASH_Tx register The value of the detector should be set to 2 times the default value, that is to keep the current Tsysclk*FLASH_Tx result equal to the default value That's it.

The following table lists the corresponding FLASH erase and write timing parameters at different frequencies:

	4M	8M	16M	24M	32M
FLASH_TNVS	0x20	0x40	0x80	0xC0	0x100
FLASH_TPGS	0x17	0x2E	0x5C	0x8A	0xB8
FLASH_TPROG	0x1B	0x36	0x6C	0xA2	0xD8
FLASH_TSERAS	0x4650	0x8CA0	0x11940	0x1A5E0	0x23280
FLASH_TMERASE	0x222E0	0x445C0	0x88B80	0xCD140	0x111700
FLASH_TPRCV	0x18	0x30	0x60	0x90	0xC0
FLASH_TSRCV	0xF0	0x1E0	0x3C0	0x5A0	0x780
FLASH_TMRCV	0x3E8	0x7D0	0xFA0	0x1770	0x1F40

Table 7-1 FLASH erasing time parameters under different frequencies
The operation steps to configure the erasing parameters when the system frequency is 8MHz are as follows:

Step1 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register sequentially to enable register rewriting.

Step2 : Write 0x40 to the FLASH_TNVS register, if the value of the register is not 0x40, then jump to

Previous.

Step3 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register sequentially to enable register rewriting.

Step4 : Write 0x2E to the FLASH_TPGS register, if the value of the register is not 0x2E, then jump to

Previous.

Page 205

Step5 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register sequentially to enable register rewriting.

Step6 : Write 0x36 to the FLASH_TPROG register, if the value of the register is not 0x36, then jump

Go to the previous step.

Step7 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register sequentially to enable register rewriting.

Step8 : Write 0x8CA0 to the FLASH_TSERASE register, if the value of the register is not 0x8CA0,

Then jump to the previous step.

Step9 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register sequentially to enable register rewriting.

Step10 : Write 0x445C0 to the FLASH_TMERASE register, if the value of the register is not 0x445C0,

Then jump to the previous step.

Step11 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register in sequence to enable register rewriting.

Step12 : Write 0x30 to the FLASH_TPRCV register, if the value of the register is not 0x30, then jump

Go to the previous step.

Step13 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register to enable register rewriting.

Step14 : Write 0x1E0 to the FLASH_TSRCV register, if the value of the register is not 0x1E0, skip

Go to the previous step.

Step15 : Write 0x5A5A and 0xA5A5 to the FLASH_BYPASS register sequentially to enable register rewriting.

Step16 : writing to register FLASH_TMRCV 0x7D0, as the value of the register is read out is not 0x7D0, then

Jump to the previous step.

Page 206

7.5 Read wait cycle

The fastest fetch frequency supported by the built-in FLASH of this device is 24MHz. When the HCLK frequency exceeds 24MHz,

A wait cycle must be inserted for CPU instruction fetching, that is, FLASH_CR.WAIT is set to 1. When inserting a wait cycle,

The CPU reads the instruction code in the FLASH memory only once every two cycles.

7.6 Erase and write protection

7.6.1 Erase and write protection bits

The entire 32kByte FLASH memory is divided into 64 Sectors, and every 4 Sectors share one erase and write protection

Bit. When the sector is protected, the erase and write operations performed on the sector are invalid, and an alarm flag bit and an interrupt signal are gen

No. When any Sector in the FLASH memory is protected, erasing and writing the Chip of the FLASH is invalid.

And generate alarm flag bit and interrupt signal.

7.6.2 PC address erase and write protection

When the CPU is running the program in FLASH, if the current PC pointer happens to fall within the address range of the sector to be erased

, Then the erasing and writing operation is invalid and generates an alarm flag and an interrupt signal. .

7.7 Register write protection

The important controller of this module shields ordinary write operations and can only be modified in the write sequence mode.

The registers that need to be changed by writing sequence are as follows:

FLASH_TNVS, FLASH_TPGS, FLASH_TPROG, FLASH_TSERASE, FLASH_TMERASE,

FLASH_TPRCV, FLASH_TSRCV, FLASH_TMRCV, FLASH_CR, FLASH_SLOCK.

The registers that can be changed without writing sequence are as follows:

FLASH_ICLR, FLASH_BYPASS.

The specific operation steps to modify the register value by writing sequence are as follows:

Step1 : Write 0x5A5A to the FLASH_BYPASS register.

Step2 : Write 0xA5A5 to the FLASH_BYPASS register.

Step3 : Write the target value to the register to be modified.

Step4 : Verify whether the current value of the register to be modified is the same as the target value, if not, jump to Step1.

Step5 : Perform other operations.

Notice:

– Write 0x5a5a and write 0xa5a5 between these two operations, no write operation can be inserted, and it cannot be interrupted by interrupts.

Otherwise, the Bypass sequence will be invalid and the sequence 0x5a5a-0xa5a5 needs to be rewritten.

7.8 Register

Base address: 0x4002 0000

register	Offset address	describe
----------	----------------	----------

FLASH_TNVS	0x00	Tnvs time parameters
FLASH_TPGS	0x04	Tpgs time parameter
FLASH_TPROG	0x08	Tprog time parameter
FLASH_TSERASE	0x0C	Tserase time parameter
FLASH_TMERASE	0x10	Tmerase time parameter
FLASH_TPRCV	0x14	Tprcv time parameter
FLASH_TSRCV	0x18	Tsrcv time parameter
FLASH_TMRCV	0x1C	Tmrcv time parameter
FLASH_CR	0x20	Control register
FLASH_IFR	0x24	Interrupt flag register
FLASH_ICLR	0x28	Interrupt flag clear register
FLASH_BYPASS	0x2C	0x5a5a-0xa5a5 Bypass sequence register
FLASH_SLOCK	0x30	Sector erase and write protection register

7.8.1 TNVS parameter register (**FLASH_TNVS**)

Offset address: 0x00

Reset value: 0x0000 0020

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
Reserved															
R															
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TNVS															
RW															

Bit	mark	Function description
31:9	Reserved	
8:0	TNVS	Calculation formula: TNVS = 8*HCLK, the unit of HCLK is MHz. Modify the register values are detailed in 7.2. 4MHz example: TNVS = 8*4 = 32.

7.8.2 TPGS parameter register (**FLASH_TPGS**)

Offset address: 0x04

Reset value: 0x0000 0017

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
Reserved															
R															
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPGS															
RW															

Bit	mark	Function description
-----	------	----------------------

31:8 RESERVED

7:0 TPGS Calculation formula: TPGS = 5.75*HCLK, the unit of HCLK is MHz. Modify the register values are detailed in. 7 [7](#).

4MHz example: TPGS = 5.75*4 = 23.

7.8.3 TPROG parameter register (**FLASH_TPROG**)

Offset address: 0x08

Reset value: 0x0000 001B

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
RW															

Bit mark Function description

31:8 Reserved

7:0 TPROG Calculation formula: TPROG = 6.75*HCLK, the unit of HCLK is MHz. Modify the register values are detailed in. 7 [7](#).

4MHz example: TPROG = 6.75*4 = 27.

7.8.4 TSERASE register (**FLASH_TSERASE**)

Offset address: 0x0C

Reset value: 0x0000 4650

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
TSERASE															
RW															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSERASE															
RW															

Bit mark Function description

31:18 Reserved

17:0 TSERASE Calculation formula: TSERASE = 4500*HCLK, the unit of HCLK is MHz. Modify the register values are detailed in. 7 [7](#).

4MHz example: TSERASE = 4500*4 = 18000.

7.8.5 TMERASE parameter register (**FLASH_TMERASE**)

Offset address: 0x10

Reset value: 0x000222E0

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMERASE															
RW															

Bit	mark	Function description
-----	------	----------------------

- | | | |
|-------|--|--|
| 31:21 | Reserved | |
| 20:0 | TMERASE calculation formula: TMERASE = 35000*HCLK, the unit of HCLK is MHz. Modify the register values are detailed in 7 7.2 . | |
| | 4MHz example: TMERASE = 35000*4 = 140000. | |

Page 211**7.8.6 TPRCV parameter register (FLASH_TPRCV)**

Offset address: 0x14

Reset value: 0x0000 0018

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPRCV															
RW															

Bit	mark	Function description
-----	------	----------------------

- | | | |
|-------|---------------------------------|--|
| 31:12 | Reserved | |
| 11:0 | TPRCV | Calculation formula: TPRVC = 6*HCLK, the unit of HCLK is MHz. Modify the register values are detailed in 7 7.2 . |
| | 4MHz example: TPRCV = 6*4 = 24. | |

7.8.7 TSRCV parameter register (FLASH_TSRCV)

Offset address: 0x18

Reset value: 0x0000 00F0

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSRCV															
RW															

Bit	mark	Function description
31:12	Reserved	
11:0	TSRCV	Calculation formula: TSRCV = 60*HCLK, the unit of HCLK is MHz. Modify the register values are detailed in 7 7.2 . 4MHz example: TSRCV = 60*4 = 240.

Page 212**7.8.8 TMRCV parameter register (FLASH_TMRDV)**

Offset address: 0x1C

Reset value: 0x0000 03E8

31	30	29	28	27	26	25	twentynine	twentynine	threentwenty	twentynine	twentynine	on	20	19	18	17	16	
Reserved																		
R																		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reserved																		
TMRCV																		
RW																		

Bit	mark	Function description
-----	------	----------------------

31:13 RESERVED

12:0 TMRCV Calculation formula: TMRCV = 250*HCLK, the unit of HCLK is MHz. Modify the register values are detailed in 7 [7.2](#).
4MHz example: TMRCV = 250*4 = 1000.**7.8.9 CR register (FLASH_CR)**

Offset address: 0x20

Reset value: 0x0000 0200

31	30	29	28	27	26	25	twentynine	twentynine	threentwenty	twentynine	twentynine	on	20	19	18	17	16	
Reserved																		
R																		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Reserved																		
IE																		
BUSY																		
Res																		
WAIT																		
OP																		
RW																		

Bit	mark	Function description
-----	------	----------------------

31:7 Reserved

6:5 IE IE[6]: FLASH erase and write protected address interrupt enable; 0: disable; 1: enable

IE[5]: FLASH erase and write PC value interrupt enable; 0: disable; 1: enable

4 BUSY Idle/busy flag bit; 0: idle state; 1: busy state;

3 Reserved

2 WAIT Read FLASH wait cycle; 0: 0 wait cycle; 1: 1 wait cycle

1:0 OP FLASH operation; 00: read; 01: program; 10: sector erase; 11: chip erase

The method of modifying the value of this register is detailed in [7.7](#).

Page 213**7.8.10 IFR register (FLASH_IFR)**

Offset address: 0x24

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twentynine	twentynine	threentwenty	twentynine	twentynine	onethirty	19	18	17	16
Reserved																
R																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reserved																
R																
IF1 IF0																
RO RO																

Bit	mark	describe
31:2	Reserved	
1	IF1	Erase protection alarm interrupt flag bit
0	IF0	Erase PC address alarm interrupt flag bit

7.8.11 ICLR register (FLASH_ICLR)

Offset address: 0x28

Reset value: 0x0000 000F

31	30	29	28	27	26	25	twentynine	twentynine	threentwenty	twentynine	twentynine	onethirty	19	18	17	16
Reserved																
R																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reserved																
ICLR ICLR																
1 0																
RW0 RW0																

Bit	mark	Function description
31:4	Reserved	
3:2	Reserved	Write invalid, read as 0x3
1	ICLR1	Clear the protection alarm interrupt flag bit; write 0 to clear; write 1 is invalid;
0	ICLR0	Clear the PC address alarm interrupt flag bit; write 0 to clear; write 1 is invalid;

7.8.12 BYPASS register (**FLASH_BYPASS**)

Offset address: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four	twenty five	twenty six	twenty seven	twenty eight	one	two	19	18	17	16
Reserved																	
R																	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
BYSEQ																	
WO																	

Bit	mark	describe
31:16	Reserved	
15:0	BYSEQ	<p>Before modifying the register of this module, the sequence 0x5a5a-0xa5a5 must be written to the BYSEQ[15:0] register. Every time you write to</p> <p>After the bypass sequence, the register can only be modified once. If you need to modify the register again, you must enter the sequence 0x5a5a-0xa5a5 again</p> <p>List. See 7.7 for details ▲</p>

7.8.13 SLOCK register (**FLASH_SLOCK**)

Offset address: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16
Reserved														
R														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
SLOCK														
RW														
Bit	mark	describe												
31:16	Reserved													
15:0	SLOCK	Sector erase and write protection bit; 0: erase and write are not allowed; 1: erase and write are allowed												
		SLOCK[0] corresponds to: Sector0-1-2-3												
		SLOCK[1] corresponds to: Sector4-5-6-7												
		SLOCK[2] corresponds to: Sector8-9-10-11												
		SLOCK[3] corresponds to: Sector12-13-14-15												
		...												
		SLOCK[15] corresponds to: Sector60-61-62-63												

8 RAM controller (RAM)

8.1 Overview

This product contains a static SRAM with a capacity of 2K/4K byte, which supports byte (8bits), half-word (16bits), Word (32bits) three read and write operations. Read and write operations can be performed at the HCLK frequency without waiting for the cycle. also, The controller also supports parity check, which can perform parity check on each byte of SRAM data and generate parity. The verification error is interrupted.

8.2 Function description

Read and write bit width

The controller supports read and write operations with three bit widths: Byte (8bits), Half-word (16bits), and Word (32bits).

The address of Byte operation must be aligned according to Byte, and the target address of Half-word operation must be aligned according to Half-word (The lowest bit of the address is 0), the address for Word operation must be aligned with Word (the lowest two bits of the address are 0). if

The target address of the read and write operation is not aligned according to the bit width, the operation is invalid, and the system will generate a Hard I
Interrupted by an error.

Parity check

This controller supports parity check of SRAM data. After reset, this function is turned off by default. When parity is turned on

After the function, when writing data to the SRAM, the parity check is performed on each Byte data, and the 1bit check value is added to the 8bits data is stored in SRAM together. When reading data to SRAM, the controller will read 8bits data and

1bit check value, and do parity check. If check error occurs, set the parity error flag bit and enable the interrupt.

In this case, an error interrupt will be generated.

Notice:

- When parity check is enabled, SRAM must be initialized before reading SRAM data, otherwise it may be touched by mistake
 - Send parity alarm flag or interrupt.

8.3 Register

Base address: 0x4002 0400

register	Offset address	describe
RAM_CR	0x00	Control register
RAM_ERRADDR	0x04	Error address register
RAM_IFR	0x08	Error interrupt flag register
RAM_ICLR	0x0C	Error interrupt flag clear register

8.3.1 Control Register (RAM CR)

Offset address: 0x00

Reset value: 0x0000 0000

Bit	mark	Function description
31:2 RESERVED		
1	IE	Error alarm interrupt enable signal; 1: enable alarm interrupt, 0: close alarm interrupt;
0	CHKEN	Parity check enable signal; 1: enable parity check, 0: disable parity check;

Page 218**8.3.2 Parity error address register (RAM_ERRADDR)**

Offset address: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRADDR															
R								RO							

Bit	mark	Function description
31:12 Reserved		
11:0	ERRADDR	12bits parity error byte address

8.3.3 Error interrupt flag register (RAM_IFR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR								RO							
R								RO							

Bit	mark	Function description
31:1	RESERVED	
0	ERR	Parity error flag

Page 219**8.3.4 Error interrupt flag clear register (**RAM_ICLR**)**

Offset address: 0x0C

Reset value: 0x0000 0001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
ERR															
CLR															
RW0															

Bit	mark	Function description
31:1	Reserved	
0	ERRCLR	Error interrupt flag clear bit; write 1: invalid, write 0: clear

Page 220

9 Basic timer (TIM0/1/2)

9.1 Introduction to basic timers

The basic timer includes three timers Timer0/1/2. Timer0/1/2 have exactly the same functions. Timer0/1/2 are synchronous Timer/counter, can be used as a 16-bit timer/counter with auto-reload function, or as a 32-bit timer/counter without reload Function timer/counter. Timer0/1/2 can count external pulses or realize system timing.

Base timer

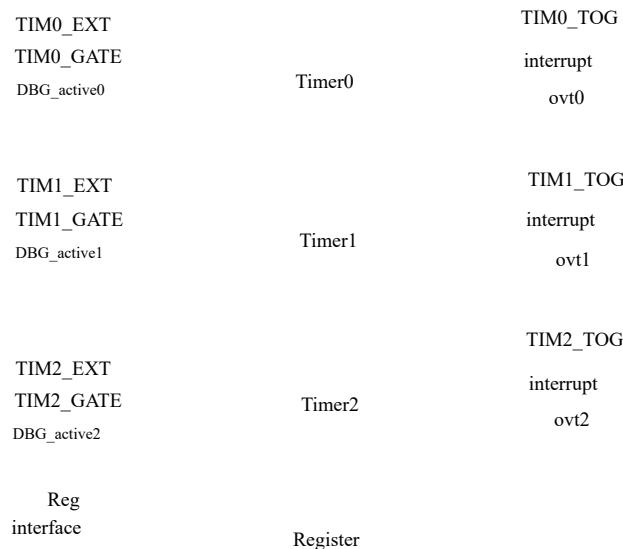


Figure 9-1 Block diagram of Base Timer

9.2 Base Timer function description

Timer0/1/2 Each timer/counter has an independent control start signal, as well as external input clock and gate control signals.

Timer0/1/2 use EXT, GATE for counting function, EXT is used for the external input clock signal of the counter,

GATE is used for the effective level counting enable signal. When the gate control function is enabled, if and only when the external input GATE power The counter will count only when the level is valid, otherwise the counter is in the holding state. The gate control can be controlled by CR.gate.

The gating function is turned off by default. The gate control level selection uses CR.GATE_P control. The default high level is the gated effective level; After it is set to 1, the gated low level is the effective level.

TIM0/1/2 use PCLK and GATE for timing function, and PCLK is used for the internal input clock signal of the timer.

No., GATE can be used as an effective level timing enable signal. When the gate control function is enabled, if and only when the external input GATE

When the level is valid, the timer will count, otherwise the timer will be in the stop state of the timer counter. Gating enable use

CR.Gate control. The gating function is turned off by default. The gate control level selection uses CR.Gate_P control. The default high level is

Gating effective level; after setting to 1, the gating effective level is low. The timing function can be configured with pre-frequency division. CR.PRS

Control the frequency division ratio.

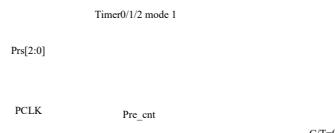
PRS [2:0]	000	001	010	011	100	101	110	111
Frequency divisibn ratio	2	4	8	16	32	64	256	

The timer/counter of TIM0/1/2 supports two working modes, by setting MD in the timer control register (CR)

Select the working mode. Mode 1 is a 32-bit free counting mode. Mode 2 is a 16-bit reload mode.

32-bit free counting mode, when the count reaches the maximum 0xFFFFFFFF, an interrupt is generated after overflow, and the timer/counter becomes 0X00000000; Then continue counting; 16-bit reload mode, overflow after counting to the maximum 0xFFFF, generate interrupt,

The value of the timer/counter is loaded as the value of ARR, and then continues to count up.



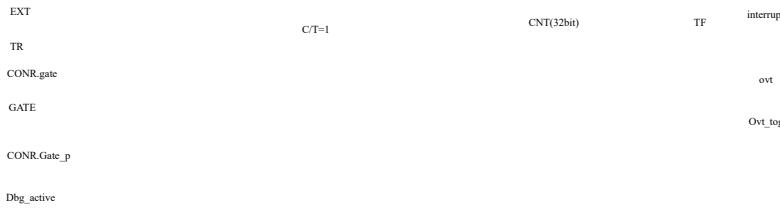


Figure 9-2 Block diagram of Timer Mode 1

In mode 2 heavy load mode, the software processing speed needs to be considered when the timing time is set to be small, otherwise the interrupt will co

Loss of interruption caused by delay in processing.

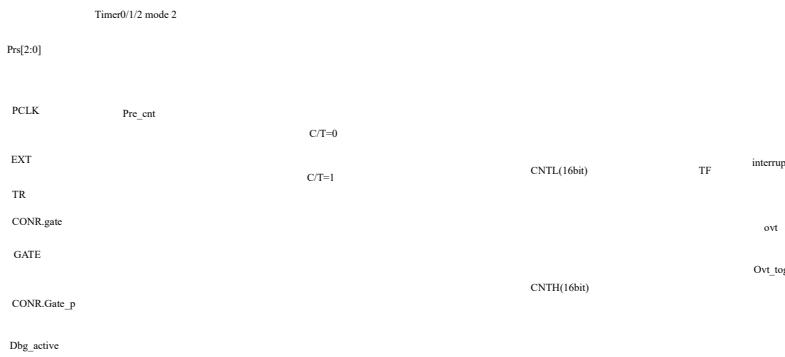


Figure 9-3 Block diagram of Timer Mode 2

When the corresponding timer TR is set to 1, the timer starts to run. Mode 1 is a 32-bit timer/counter, which will start from

The initial value T set by the register starts to count, and an overflow interrupt is generated when the count reaches the maximum. Then continue countir

Formula 2 is a 16-bit reload timer/counter. After startup, it starts to count up from the initial value of the register CNT, and the count reaches the maximu

After an interrupt is generated after 0xFFFF, the value of the register ARR is reloaded to the counter CNT and continues to count up.

9.2.1 Counting function

The counting function is used to determine the number of occurrences of an event. In the counting function, the counter is

The falling edge of the clock accumulates once. The input signal is sampled by the internal PCLK, so the external input clock frequency cannot exceed

Pass the system's Pclk clock. When the count reaches the maximum value, it will overflow and generate an interrupt. The interrupt flag needs to be clea

Timer0/1/2 mode 1 (max=0xFFFF_FFFF)

PCLK

TR

Count(32) T T+1 T+2 T+3 T+4 ...

max 0 1 2

max 0 1 2

max 0 1 2

ovt



Figure 9-4 Timing diagram of Mode 1

9.2.2 Timing function

The timing function is used to generate interval timing. In the timing function, the timer has a pre-frequency division, and the timer is in each pre-frequency mode. A clock of accumulates once, and when the count reaches the maximum value, it will overflow and generate an interrupt. The interrupt flag needs to be cleared.



Figure 9-5 Timing diagram of Mode 2 (prescaler is set to 2)

9.2.3 Buzzer function

The function of driving the Buzzer can be realized through the flip output function of the timer. When CR.TOG_EN is 1, TOG, TGN output is reversed. Set CR.TOG_EN to 0 to set port TOG and TGN output at the same time.

Is 0. When the count clock is 4M, the Buzzer output timer reload mode with different frequencies is configured as follows:

Buzzer frequency	Counter cycle	counter count value	counter reload value	CNT initial value	ARR reload value
1000Hz	0.5ms	2000	63536	0xF830	0xF830
2000Hz	0.25ms	1000	64536	0xFC18	0xFC18
4000Hz	0.125ms	500	65036	0xFE0C	0xFE0C

Page 225

9.3 Base Timer interconnection

9.3.1 GATE interconnection

The GATE input can be directly input from the port, or the RX signal of the UART can be input; it can also be configured as VC

The input is used as the GATE signal. The GATE of Timer0/1/2 can be configured.

Through the internal interconnection configuration, the automatic identification of UART baud rate can be realized, and the comparison output of VC c

The pulse width can realize external control counting.

The configuration selection RX input is controlled by the GPIO_CTRL register, and the VC control is controlled by the VC control register.

Only one of port selection, UART input selection and VC input selection as gate control input can be selected to be valid.

9.3.2 Toggle output interconnection

The inversion output of TIM0 tog0 to the internal module UART0, which controls the baud rate of UART0; the inversion of TIM1

Output tog1 to the internal module UART1 to control the baud rate of UART1; the inversion output of TIM0/1/2 is also output

Out to the port, it can drive the Buzzer to control the buzzer.

Page 226**9.4 Base Timer register description**

x=0,1,2;

Base Timer base address 0X40000C00

Timer	Offset address	describe
TIM0	0x00	TIM0 offset address
TIM1	0x20	TIM1 offset address
TIM2	0x40	TIM2 offset address

register	Offset address	describe
TIMx_ARR	0X000	TIM0/1/2 reload register
TIMx_CNT	0X004	TIM0/1/2 16-bit mode count register
TIMx_CNT32	0X008	TIM0/1/2 32-bit mode count register
TIMx_CR	0X00C	TIM0/1/2 control register
TIMx_IFR	0X010	TIM0/1/2 interrupt flag
TIMx_ICLR	0X014	TIM0/1/2 interrupt clear register

Table 9-1 Base Timer register list

9.4.1 16 -bit mode reload register (TIMx_ARR)

Offset address: 0x000

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twent	twent	twent	twent	on	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															
R/W															

Bit	symbol	describe
31:16	Reserved	Reserved bit, read as 0
15:0	ARR	Timer mode 2 reload register

Page 227**9.4.2 16 -bit mode count register (TIMx_CNT)**

Offset address: 0x004

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16
Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
CNT[15:0]														
R/W														

Bit symbol describe

31:16 Reserved Reserved bit, read as 0

15:0 CNT Timer mode 2 count value register

9.4.3 32 -bit mode count register (TIMx_CNT32)

Offset address: 0x008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16
CNT32[31:16]														
R/W														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
CNT32[15:0]														
R/W														

Bit symbol describe

31:0 CNT32 Timer mode 1 count value register

9.4.4 Control Register (TIMx_CR)

Offset address: 0x00C

Reset value: 0x0000 0008

31:11	10	9	8	7	6:4	3	2	1	0
Reserved	IE	GATE_P	GATE	Reserved	PRS	TOG_EN	CT	MD	TR
	RW	R/W	R/W		R/W	R/W	R/W	R/W	R/w

Bit	symbol	describe
31:11	Reserved	Reserved bit, read as 0
10	IE	Interrupt enable control, enable interrupt after writing 1
9	GATE_P	Port GATE polarity control, the default high-level gate is effective, and low-level is effective when set to 1
8	GATE	Timer gating 0: No gate control, the timer works when TR=1; 1: Only work when port GATE is valid and TR=1;
7	Reserved	Reserved bit, read as 0
6:4	PRS	TIM pre-frequency selection. 000:1; 001:2; 010:4; 011:8; 100:16; 101:32; 110:64; 111:256;
3	TOG_EN	TOG output enable 0: TOG and TOGN output 0 at the same time 1: TOG, TOGN output signals with opposite phases. Can be used by buzzer.
2	CT	Counter/timer function selection 0: Timer function, the timer is counted by PCLK. 1: Counter function, the counter counts by the falling edge of the external input. The external input is taken by PCLK Similarly, the external input clock frequency should be lower than 1/2 of the sampling clock.
1	MD	Timer working mode 0: Mode 1 32-bit counter/timer 1: Mode 2 automatic reload 16-bit counter/timer
0	TR	Timer operation control 0: The timer stops 1: Timer running

9.4.5 Interrupt Flag Register (TIMx_IFR)

Offset address: 0x010

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved								
								TF RO

Bit	symbol	describe
31:1	REV	Reserved bit, read as 0
0	TF	Interrupt flag, set by hardware. Write to clear the register to clear

9.4.6 Interrupt flag clear register (**TIMx_ICLR**)

Offset address: 0x014

Reset value: 0x0000 0001

31:8	7	6	5	4	3	2	1	0
Reserved								
								TFC W0

Bit	symbol	describe
31:1	Reserved	reserved bit, read as 0
0	TFC	Interrupt flag is cleared, write 0 to clear, write 1 to have no effect

10 Low-power timer (**LPTIM**)

10.1 Introduction to **LPTimer**

LPTimer is an asynchronous 16-bit timer/counter. After the system clock is turned off, it can still pass the internal low-speed RC or External low-speed crystal oscillator timing/counting. Wake up the system in low-power mode through interrupts.

EXT	Low Power Timer	TOG
GATE		TOGN
Dbg_active		interrupt

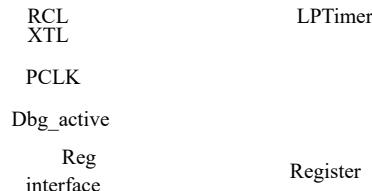


Figure 10-1 LPTimer block diagram

10.2 LPTimer function description

LPTimer supports 2 working modes, each timer/counter has an independent control start signal, and external Input clock, gate control signal.

LPTimer uses EXT, GATE for counting function, EXT is used for counter's external input clock signal, Gate is used for the effective level counting enable signal.

The timer of LPTimer supports two working modes, which can be selected by setting MD in the timer control register (CR).

Select the working mode. Mode 1 is a 16-bit free counting mode. Mode 2 is a 16-bit reload mode.

When LPTimer starts, it will automatically load the value of the reload register ARR into the counter.

LPTimer can choose three clocks as the timer clock, which can be selected by the control register CR.TCK_SEL. default

Select PCLK. Clock selection as table:

TCK_SEL	00	01	10	11
Timer clock	PCLK	PCLK	XTL	RCL

Read timer count value read after synchronization. Reads are synchronized

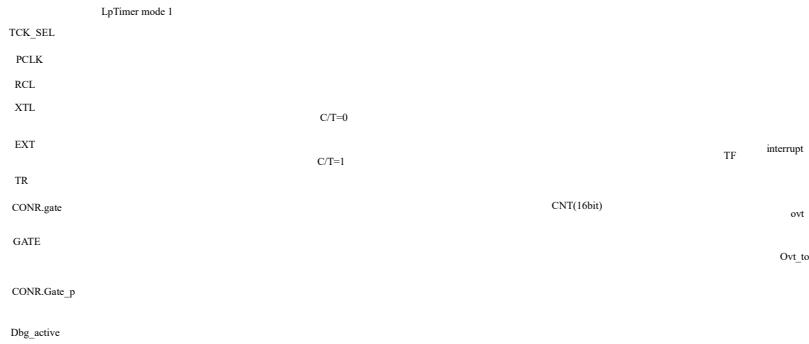


Figure 10-2 LPTimer mode 1

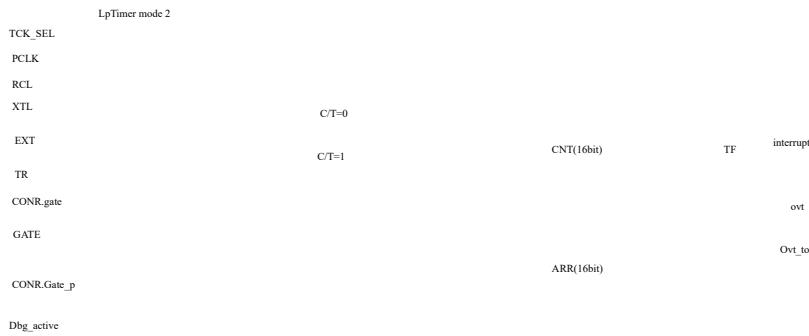
Page 232

Figure 10-3 LPTimer mode 2

When the corresponding timer TR is set to 1, the timer starts to run. The counter starts counting from the set value, and counts up. An overflow interrupt is generated after the maximum 0xFFFF. Mode 1 After the interrupt is generated, the counter is cleared and continues to count up. After the interrupt is generated, the value of the register ARR is reloaded into the counter and counts up. The initial value of the counter CNT is starting at 0. The timer is automatically loaded by ARR.

Since LPTimer is an asynchronous timer, the timer interrupt is synchronized from the Timer clock domain to pclk pre-reload mode.

When the timer value is set greater than 0Xffff, the interrupt will be lost. It is recommended to use the interrupt function if the value of the reload register is set to 0XFFFC. There is no such limitation if interrupts are not used.

10.2.1 Counting function

The counting function is used to determine the number of occurrences of an event. In the counting function, the counter is

The falling edge of the clock accumulates once. The input signal is sampled by the internal count clock, so the external input clock frequency cannot exceed the system's count clock. When the count reaches the maximum value, it will overflow and generate an interrupt.

10.2.2 Timing function

The timing function is used to generate interval timing. In the timing function, the timer accumulates once per clock and counts to the maximum value. The value will overflow and an interrupt will be generated.

10.3 LPTimer interconnection

10.3.1 GATE interconnection

The GATE input can be directly input from the port, or the RX signal of the UART can be input; it can also be configured as VC input.

The input is used as the GATE signal. The GATE of LPTIM can be configured as follows.

Through the internal interconnection configuration, the automatic identification of UART baud rate can be realized, and the comparison output of VC can be realized.

The pulse width can realize external control counting.

The UART selection control register is in the port control register GPIO_CTRL4, and the VC output control register is in the port control register.

VC control module.

10.3.2 EXT interconnection

EXT input can be input directly from the port, or it can be configured as VC input as EXT signal. LPTIM

EXT can be configured as follows.

Through internal interconnection configuration, VC pulse count can be measured. The VC output control register is in the VC control module.

10.3.3 Toggle output interconnection

The flip of LPTimer is output to the port, which can drive the Buzzer to control the buzzer.

Page 234

10.4 LPTimer register description

Base address 0X40000C00

register	Offset address	describe
CNT	0X060	LPTimer count value read-only register
ARR	0X064	LPTimer reload register
CR	0X06C	LPTimer control register
IFR	0X070	LPTimer interrupt flag
ICLR	0X074	LPTimer interrupt clear register

Table 10-1 LPTimer register list

Page 235

10.4.1 Counter count value register (LPTIM_CNT)

Offset address: 0x060

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty fourty thwenty twenty on�	19	18	17	16
Reserved											
15	14	13	12	11	10	9	8	7	6	5	4
CNT[15:0]											

RO

Bit	symbol	describe
31:16	Reserved	Reserved bit, read as 0
15:0	CNT	Low-power timer count value read-only register. When the timer is started, the initial value of CNT is automatically loaded by ARR load.

10.4.2 Reload register (LPTIM_ARR)

Offset address: 0x064

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty fourty thwenty twenty on�	19	18	17	16
Reserved											
15	14	13	12	11	10	9	8	7	6	5	4
ARR[15:0]											

R/W

Bit	symbol	describe
31:16	Reserved	Reserved bit, read as 0
15:0	ARR	Low-power timer reload register It is necessary to read CR.WT_FLAG before writing ARR, and only when WT_FLAG is 1, can it be written data input. WT2_FLAG will go low after writing the ARR register.

Page 236

10.4.3 Control Register (LPTIM_CR)

Offset address: 0x06C

Reset value: 0x0000 0008

31:11	10	9	8	7	6	4:5	3	2	1	0
Reserved	IE	GATE_P	GATE	WT_FLAG	Reserved	TCK_SEL	TOG_EN	CT	MD	TR

RW R/W R/W RO Reserved R/W R/W R/W/RW/RW/R/W

Bit	symbol	describe
31:11	Reserved	Reserved bit, read as 0
10	IE	Interrupt enable control, enable interrupt after writing 1
9	GATE_P	GATE polarity control, the default high-level gate is effective, and low-level is effective when set to 1
8	GATE	Timer gating 0: No gate control, the timer works when TR=1; 1: Only work when the gate is 1 and TR=1;
7	WT_FLAG	WT, write synchronization flag 0: synchronizing, writing ARR is invalid at this time 1: Synchronization is complete, ARR can be changed at this time
6	Reserved	Reserved bit, read as 0
5:4	TCK_SEL	LPTimer clock selection 00 PCLK; 10: XTL; 11, RCL
3	TOG_EN	TOG output enable 0: TOG and TOGN output 0 at the same time 1: TOG, TOGN output signals with opposite phases. Can be used by buzzer.
2	CT	Counter/timer function selection 0: Timer function, the timer uses the clock selected by TCK_SEL to count. 1: Counter function, the counter uses the falling edge of the external input to count. Sampling clock usage The clock selected by TCK_SEL, the external input clock must be lower than 1/2 the sampling clock.
1	MD	Timer working mode 0: Mode 1 non-reload mode 16-bit counter/timer 1: Mode 2 automatic reload 16-bit counter/timer
0	TR	Timer run control bit 0: The timer stops 1: Timer running

10.4.4 Interrupt Flag Register (LPTIM_IFR)

Offset address: 0x070

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved							TF	RO

Bit	symbol	describe

31:1 0	Reserved reserved bit, read as 0 TF	Interrupt flag, set by hardware. Write to clear the register to clear
-----------	--	---

10.4.5 Interrupt Flag Clear Register (**LPTIM_ICLR**)

Offset address: 0x074

Reset value: 0x0000 0001

31:8	7	6	5	4	3	2	1	0
Reserved								TFC
								W0

Bit	symbol	describe
31:1	Reserved	reserved bit, read as 0
0	TFC	Interrupt flag is cleared, write 0 to clear, write 1 to have no effect

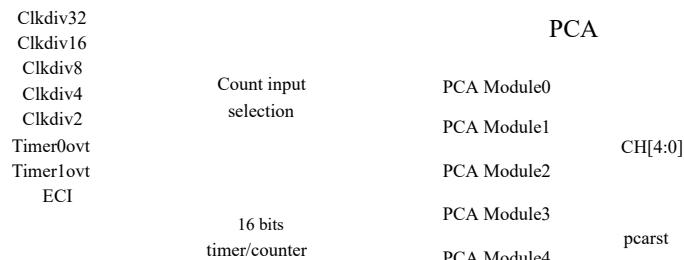
11 Programmable Counting Array (**PCA**)

11.1 Introduction to PCA

PCA (Programmable Counter Array) supports up to 5 16-bit capture/compare

Module. The timer/counter can be used as a common clock count/event counter capture/compare function. PCA

Each of the modules can be independently programmed to provide input capture, output comparison or pulse width modulation. in addition Module 4 has an additional watchdog timer mode.



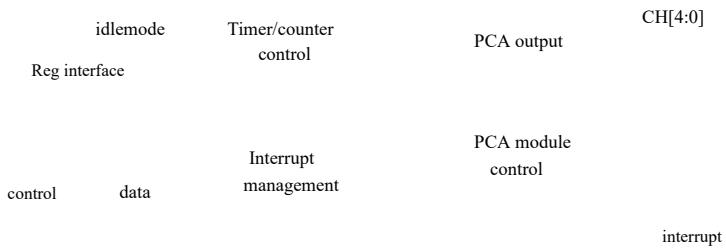


Figure 11-1 Overall block diagram of PCA

11.2 PCA function description

Each module can be configured to work independently, there are three working modes: edge trigger capture, output comparison, 8-bit pulse Wide modulation. Each module has its own function register in the system controller, these registers are used for configuration How the module works and exchange data with the module.

Each compare/capture module consists of a compare/capture register set (CCAPx), and a 16-bit comparator, and Various logic gate control components. The register group is used to store time or number of times, for external trigger capture conditions, or internal Trigger comparison conditions. In the PWM mode, the register (CCAPxL) is used to control the duty cycle of the output waveform.

Each module can be independently programmed to operate in any of the following modes:

- The 16-bit capture mode is triggered on the rising edge, falling edge or any edge.
- Comparison mode: 16-bit software timer, 16-bit high-speed output, 16-bit watchdog timer (module 4) or 8-bit Pulse width modulation.
- Not started.

The compare/capture module mode register (CCAPMx) determines the corresponding operating mode. For the compare/capture module When programming, they are based on a common time count. The timer/counter is turned on and off by the CCON.CR bit. It can control the operation of PCA timer/counter. Capture in a compare/capture module, software timer, high-speed output, Set the compare/capture flag (CCON.CCFx) of the module, and generate a PCA interrupt request, if the corresponding enable Bits are set in the CCAPMx register. The CPU can read and write the CCAPx register at any time.

11.2.1 PCA Timer / Counter

The special function register of CNT can be used as a 16-bit timer/counter. This is a 16-bit up Counting counter. If the CMOD.CFIE bit is set to "1", the hardware will automatically set PCA when CNT overflows Overflow flag (CCON.CF) and generate PCA interrupt request. CMOD.CPS[2:0] Three bits select eight signals Input to the timer/counter.

The system clock PCLK divided by 32.

System clock PCLK divided by 16.

The system clock PCLK divided by 8.

The system clock PCLK divided by 4.

The system clock PCLK divided by 2.

Page 240

Timer 0 overflow. Every time the timer 0 counts overflow, CNT is incremented, thus providing PCA

Variable programming frequency input.

Timer 1 overflow. Every time the timer 1 count overflows, CNT is incremented, which provides PCA

Variable programming frequency input.

ECI. The CPU samples the PCA ECI every 4 PCLK clock cycles. When the result of each sampling is from

When high changes to low, CL automatically increases by 1, so the highest ECI input frequency cannot be higher than the system clock PCLK 1/8 to meet sampling requirements.

Set the running controller (CCON.CR) to start the PCA timer/counter. When CMOD.CIDL is set to "1",

The PCA timer/counter can continue to run in idle mode. The CPU can read the value of CNT at any time,

But when the counting is started (CCON.CR=1), in order to prevent counting errors, CNT is forbidden to write.

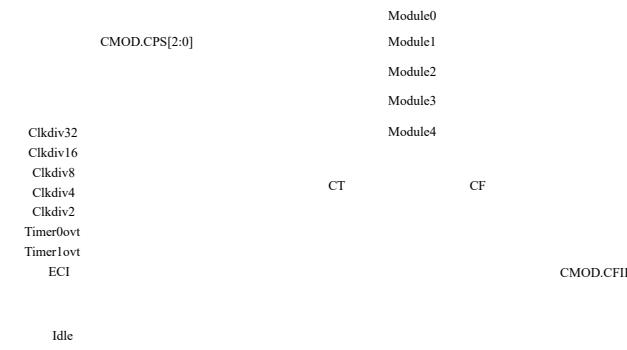


Figure 11-2 PCA counter block diagram

Page 241**11.2.2 PCA capture function**

The PCA capture mode provides 5 PCA functions to measure pulse period, pulse width, duty cycle and phase difference.

The level transition on the pin causes the PCA to capture the value of the PCA counter/timer and load it into the corresponding mode.

Block 16-bit capture/compare register (CCAPx). The CCAPMx.CAPP and CCAPMx.CAPN bits are used

Select the type of level change for trigger capture: low level to high level (positive edge), high level to low level (negative edge) or

Any change (positive or negative edge). When a capture occurs, the capture/compare flag (CCFn) in CCON is set to

Logic '1' and generate an interrupt request (if CCF interrupt is enabled). When the CPU turns to the interrupt service routine,

CCFn bit cannot be automatically cleared by hardware, user software writes INTCL register to clear this flag bit. Need to go up and down

Edge capture at the same time, the recommended process: first enable an edge capture, after the capture occurs, use it in the interrupt service routine.

Can another capture, and close the previous capture interrupt. Switch in turn.

The resolution is equal to the timer/counter clock. The input signal must maintain at least 2 during high or low level

The clock cycle to ensure that the input signal can be recognized by the hardware.

The CPU can read or write CCAPx registers at any time.

Capture settings:

When it is necessary to capture on the external rising edge, CCPMx.CAPP = "1" and CCAPMx.CAPN = "0"

When it is necessary to capture on the external falling edge, CCPMx.CAPP = "0" and CCAPMx.CAPN = "1"

When it is necessary to capture on the external rising and falling edges, CCPMx.CAPP = "1" and CCAPMx.CAPN = "1"

When you need to capture the external rising and falling edges at the same time and need to know the captured edge, first set

CCPMx.CAPP = "1", after the rising edge capture occurs, switch to the falling edge capture in the interrupt service subroutine,

Set CCAPMx.CAPN = "1", and clear CCAPMx.CAPP = "0". Set it to after the next capture

Rising edge capture, switch the captured edge in turn.

Notice:

- Subsequent captured values from the same module will overwrite existing captured values. In order to maintain the captured value, in the interrupt :

Save it in RAM in the program. This operation must be completed before the next event occurs, otherwise

The previous capture sample value will be lost.

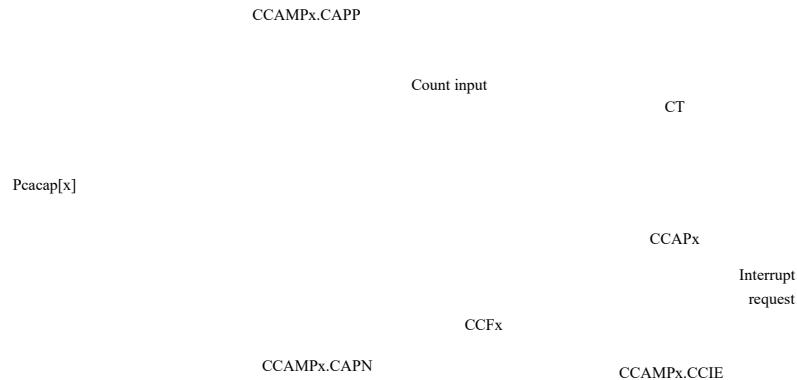
Page 242

Figure 11-3 PCA capture function block diagram

Page 243

11.2.3 PCA comparison function

The comparison function provides five modules with the following functions, timer, event counter, and pulse width modulation. Four modes Use comparison function: 16-bit software timer mode, high-speed output mode, WDT mode and PWM mode. in front

Among the three functions, the compare/capture module compares the value of the 16-bit PCA timer/counter with preloaded into the module 16-bit value in the CCAPx register. In the PWM mode, the PCA module continuously sets the PCA timer /Counter low byte register (CNT) is compared with an 8-bit value in the CCAPxL module register. Every 4 clock cycles are compared once, that is, it matches the clock rate of the fastest PCA timer/counter.

Set the CCAPMx.ECOM bit to select the compare function of the module.

To use the module in comparison mode correctly, please follow the general procedure below:

Select the operating mode of the PCA module

Select the input signal of the PCA timer/counter.

The comparison value is loaded into the comparison/capture register pair of the module.

Set the PCA timer/counter operation control bit.

An interrupt is generated after a match, and the compare/capture flag of the module is cleared.

11.2.3.1 16 -bit software counting mode

To set the 16-bit software timer mode of a compare/capture module, you need to set CCAPMx.ECOM and

CCAPMx.MAT bit. Once between PCA timer/counter and compare/capture register (CCAPx)

A match has occurred, which will set the module's compare/capture flag (CCON.CCFx). This will generate an interrupt request,

If the corresponding interrupt enable bit (CCAPMx.CCIE) is set. Interrupt due to compare/capture flag not cleared by hardware

When processing, the user must clear the software flag. In the interrupt service routine, a new 16-bit comparison value can be written

Input compare/capture register (CCAPx).

Note: When updating these registers, in order to prevent invalid matching from happening, the user software should write CCAPxL first,

Write CCAPxH afterwards. Once written to CCAPxL, the ECOMx bit of the disable compare function will be cleared, while writing

CCAPxH will set the ECOMx bit at the same time to re-enable the comparison function. That is when the capture/compare to PCA0

When writing a 16-bit value to the register, write the low byte first.

11.2.3.2 High-speed output mode

In the high-speed output mode, whenever the value in the PCA counter matches the module's 16-bit capture/compare register (CCAPx)

When a match occurs, the logic level on the CAP/CMP[x] pin of the PCA of the module will change. This can provide

It has higher accuracy than switching IO output, because this high-speed output will not be affected by interrupt response and affect the output frequency.

If the CPU switches the IO output, power consumption and accuracy are lacking.

To set the high-speed output mode of a compare/capture module, set CCAPMx.ECOM, CCAPMx.MAT and

CCAPMx.TOG bit. Match switching between PCA timer/counter and compare/capture register (CCAPx)

Change the CAP/CMP[x] signal of PCA and set the compare/capture flag (CCON.CCFx) of the module. Through soft

Set or clear the PCA's CAP/CMP[x] signal, the user can choose to match the switch signal from low to high or high

To low.

The user can also choose to generate an interrupt request by setting the corresponding interrupt enable bit (CCAPMx.CCIE) when

When a match occurs, an interrupt request can be generated. Because the hardware cannot clear the compare/capture flag interrupt, the user must

Clear this flag in software. If the user does not change the compare/capture register in the interrupt program, PCA does not

Re-count the comparison value, if there is a match, the next rollover will occur. In the interrupt service routine, a new 16-bit ratio

The comparison value can be written into the compare/capture register (CCAPx).

Note: In order to prevent invalid matches and update these registers, the user software should write the CCAPxL first

After CCAPxH. Write to CCAPxL to clear the ECOM bit that disables the compare function, and write to CCAPxH to set

ECOM bit, re-enable the comparison function.

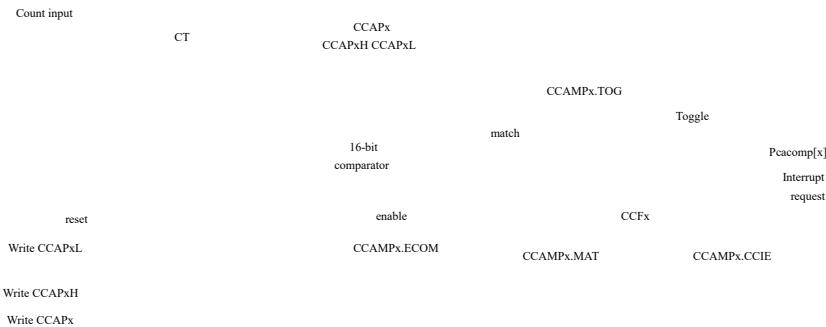


Figure 11-4 PCA comparison function block diagram

11.2.3.3 PCA module 4 of the WDT function

In addition to a WDT hardware module, PCA module 4, this product also provides a 16-bit programmable frequency

WDT. When the count value of the PCA timer/counter is compared with the value stored in module 4/capture register (CCAP4)

When matching, this mode generates a reset signal. The PCA's WDT reset signal is used as an independent reset signal.

With external reset (RST), hardware watchdog reset (WDTRST) and LVD low voltage reset, POR power on and off

Combined with electrical reset. Users can freely combine or use them individually. Module 4 is the only one with WDT mode

PCA module. When it is not set to WDT, it can be used independently in other modes.

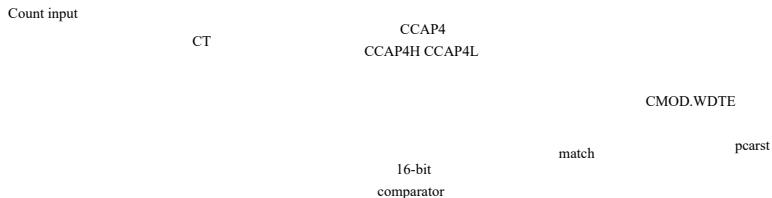




Figure 11-5 PCA WDT functional block diagram

When using PCA module 4 as WDT, you must set CCAPM4.ECOM4 and CCAPM4.MAT4 to And CMOD.WDTE. In addition, the PCA timer/counter can be set to CMOD.CPS to select different output Enter the counting frequency. Enter a 16-bit comparison value in the compare/capture register (CCAP4). In the PCA timer/counter (CNT), Enter a 16-bit initial value or use the reset value (0x0000). PCA input pulse rate multiplied by these values The difference between determines the WDT to match the running time. Set timer/counter operation control bit (CCON.CR) Start the PCA WDT. Every time there is a match, the PCA's WDT generates a reset signal. To prevent a PCA WDT To reset, the user has three choices. The comparison value CCAP4 changes regularly, so the match will never happen. Periodically change the PCA timer/counter value (CNT) so the match will never happen.

Page 246

Disable the module reset output signal by clearing the CMOD.WDTE bit before the match, and then re-enable it.

The first two options are more reliable because WDT is not disabled in the third option.

The second option is not recommended, if other PCA modules are in use, because the five modules share a common Time base. Therefore, the first option in most applications is the best.

PCA WDT configuration process:

1. Configure WDT compare/capture register PCA_CCAP4
2. Configure PCA counting register PCA_CNT
3. Configure PCA_CCAMP4 to select compare match function
4. Configure PCA_CMOD to select input clock and enable WDT function
5. Start PCA
6. Select the clear PCA WDT clear method to clear the PCA WDT before resetting the PCA WDT

Page 247**11.2.3.4 PCA 8-bit pulse width modulation function**

Pulse width modulation is a technique that uses a program to control the duty cycle, period, and phase of a waveform. 5 PCA modules are available. It can be used independently to generate pulse width modulation (PWM) output on the CAP/CMP[x] pin of the corresponding PCA, pulse. The width is 8-bit resolution. The frequency of the PWM output depends on the time base of the PCA counter/timer. Use modules. The capture/compare register CCAPxL is used to change the duty cycle of the PWM output signal. When PCA counter/timing. When the low byte (CNTL) of the device is equal to the value in CCAPxL, the output on the CAP/CMP[x] pin of PCA. The output is set to "1"; when the count value in CNTL overflows, the PCA's CAP/CMP[x] output is reset to "0". when. When the low byte CNTL of the counter/timer overflows (from 0xFF to 0x00), the value stored in CCAPxH. It is automatically loaded into CCAPxL without software intervention.

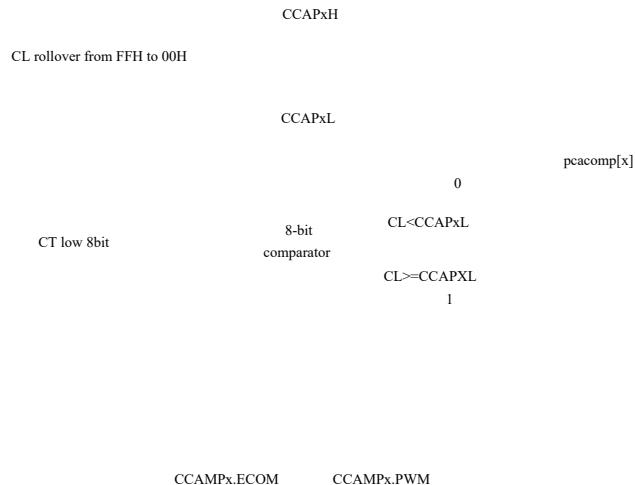


Figure 11-6 PCA PWM functional block diagram

In this mode, the value of the low byte (CNTL) of the PCA timer/counter is continuously compared with the value of the compare/capture register. The value of the low byte (CCAPxL) is compared. When CNTL < CCAPxL, the output waveform is low; when CNTL >= CCAPxL, the output waveform is high. When CNTL overflows, the system automatically loads the value of CCAPxH to

In CCAPxL, a new counting cycle is started.

The value of CCAPxL determines the duty cycle of the current cycle, and the value of CCAPxH determines the duty cycle of the next cycle. Change

The value in CCAPxL can change the duty cycle of PWM. As shown in the figure below, adjust the value of CCAPxL to 0~255

The duty cycle can be 100%~0.4%. In order to prevent glitches, it is recommended not to directly change the value of CCAPxL.

Page 248

By changing the value of CCAPxH, it will be automatically loaded into CCAPxL by the hardware in the next cycle.

CCAPxL Duty cycle	Output waveform
255	0.4
230	10
128	50
25	90
0	100

Figure 11-7 PCA PWM output waveform

To set a compare/capture module in PWM mode, you need to set CCAPMx.ECOM and

CCAPMx.PWM bit. In addition, the PCA timer/counter can be input by programming CMOD.CSP[2:0]

Count the signal frequency. Enter an 8-bit value in CCAPxL to specify the duty cycle of the first PWM waveform. exist

Inputting an 8-bit value to CCAPxH will specify the duty cycle of the second PWM waveform. Set timer/counter

The run control bit (CCON.CR) starts the PCA timer/counter.

ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	Way of working
X	1	0	0	0	0	X	Trigger capture with positive edge
X	0	1	0	0	0	X	Trigger capture with negative edge
X	1	1	0	0	0	X	Trigger capture with transition edge
1	0	0	1	0	0	X	Software timer
1	0	0	1	1	0	X	High-speed output
1	0	0	0	0	1	X	8-bit pulse width modulator

Table 11-1 PCA compare capture function module settings

11.3 PCA module and other modules interconnection and control

11.3.1 ECI interconnection

ECI input can be external input ports selected through IO MUX, or internal VC comparison

The filtered output. The VC output control register is in the VC control module.

11.3.2 PCACAP0

The capture input of channel 0 can be:

External IO MUX input port

MUX input of RX of external UART

The output after comparison and filtering of the internal VC1

The UART selection control register is in the port control register GPIO_CTRL2, and the VC output control register is in the port control register.

VC control module.

11.3.3 PCACAP1

The capture input of channel 1 can be:

External IO MUX input port

MUX input of RX of external UART

The output after comparison and filtering of the internal VC2

The UART selection control register is in the port control register GPIO_CTRL2, and the VC output control register is in the port control register.

VC control module.

11.3.4 PCACAP [4:2]

The capture input of channels 2, 3, and 4 can be:

External IO MUX input port

MUX input of RX of external UART

The UART selection control register is GPIO_CTRL2 in the port control register.

11.4 PCA register description

Base address 0X40001000

register	Offset address	describe
CCON	0X000	PCA control register
CMOD	0X004	PCA mode register
CNT	0X008	PCA count register
ICLR	0X00C	PCA interrupt clear register
CCAPM0	0x010	PCA compare/capture module 0 mode register
CCAPM1	0x014	PCA compare/capture module 1 mode register
CCAPM2	0x018	PCA compare/capture module 2 mode register
CCAPM3	0x01C	PCA compare/capture module 3 mode register
CCAPM4	0x020	PCA compare/capture module 4 mode register
CCAP0H	0X024	PCA compare/capture module 0 high 8-bit register
CCAP0L	0X028	PCA compare/capture module 0 low 8-bit register
CCAP1H	0X02C	PCA compare/capture module 1 high 8-bit register
CCAP1L	0X030	PCA compare/capture module 1 low 8-bit register
CCAP2H	0X034	PCA compare/capture module 2 high 8-bit register
CCAP2L	0X038	PCA compare/capture module 2 low 8-bit register
CCAP3H	0X03C	PCA compare/capture module 3 high 8-bit register
CCAP3L	0X040	PCA compare/capture module 3 low 8-bit register
CCAP4H	0X044	PCA compare/capture module 4 high 8-bit register
CCAP4L	0X048	PCA compare/capture module 4 low 8-bit register
CCAO	0X04C	PCA PWM and high-speed output flag register
CCAP0	0X050	16-bit register of PCA compare/capture module 0
CCAP1	0X054	16-bit register of PCA compare/capture module 1
CCAP2	0X058	16-bit register of PCA compare/capture module 2
CCAP3	0X05C	16-bit register of PCA compare/capture module 3
CCAP4	0X060	16-bit register of PCA compare/capture module 4

Table 11-2 PCA register list

11.4.1 Control Register (PCA_CCON)

Offset address: 0x000

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved	CF RO	CR R/W	Reserved	CCF4 RO	CCF3 RO	CCF2 RO	CCF1 RO	CCF0 RO

Bit	symbol	describe
31:8	Reserved reserved bit	
7	CF	PCA counter overflow flag (write invalid) When the PCA count overflows, CF is set by hardware. If the CFIE bit of the CMOD register is 1, then CF flag can generate interrupt 1: Counter overflow occurred; 0: No overflow;
6	CR	PCA counter operation control bit 1: Start the PCA counter to count 0: Turn off PCA counter counting
5	Reserved reserved bit	
4	CCF4	PCA counter module 4 compare/capture flag When a match or capture occurs, this bit is set by hardware. (Write invalid) When CCAPM4.CCIE is set, this flag will generate a PCA interrupt
3	CCF3	PCA counter module 3 compare/capture flag When a match or capture occurs, this bit is set by hardware. (Write invalid) When CCAPM3.CCIE is set, this flag will generate a PCA interrupt
2	CCF2	PCA counter module 2 compare/capture flag When a match or capture occurs, this bit is set by hardware. (Write invalid) When CCAPM2.CCIE is set, this flag will generate a PCA interrupt
1	CCF1	PCA counter module 1 compare/capture flag When a match or capture occurs, this bit is set by hardware. (Write invalid) When CCAPM1.CCIE is set, this flag will generate a PCA interrupt
0	CCF0	PCA counter module 0 compare/capture flag When a match or capture occurs, this bit is set by hardware. (Write invalid) When CCAPM0.CCIE is set, this flag will generate a PCA interrupt

Page 252**11.4.2 Mode register (PCA_CMOD)**

Offset address: 0x004

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved	CIDL	WDTE		Reserved		CPS		CFIE
	R/W	R/W				R/W		R/W

Bit	symbol	describe
31:8	Reserved reserved bit	
7	CIDL	In idle mode IDLE, whether PCA stops working 1: In sleep mode (sleep), PCA stops working 0: PCA continues to work in sleep mode (sleep)
6	WDTE	PCA WDT function enable control bit 1: Start PCA module 4 WDT function 0: Disable PCA module 4 WDT function
5:4	Reserved reserved bit	

3:1	CPS	Clock division selection and clock source selection 000: PCLK/32 001: PCLK/16 010: PCLK/8 011: PCLK/4 100: PCLK/2 101: Timer0 overflow 110: Timer1 overflow 111: ECI external clock, clock PCLK divided by four sampling
0	CFIE	PCA counter interrupt enable control signal 1: Enable interrupt 0: Turn off the interrupt

Page 253**11.4.3 Counting register (PCA_CNT)**

Offset address: 0x008

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twent	four	thre	twent	twent	on	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
CNT																
R/W																

Bit	symbol	describe
31:16	Reserved	reserved bit
15:0	CNT	Timer counter value CNT can be written only in the PCA stop state, otherwise the writing is invalid

11.4.4 Interrupt Clear Register (PCA_ICLR)

Offset address: 0x00C

Reset value: 0x0000 009Fh

31:8	7	6	5	4	3	2	1	0
Reserved	CF	Reserved		CCF4	CCF3	CCF2	CCF1	CCF0
	W0			W0	W0	W0	W0	W0

Bit	symbol	describe
31:8	Reserved	reserved bit
7	CF	PCA counter overflow flag is cleared (software writes 0 to clear, writes 1 is invalid), the read value is 1
6:5	Res.	Reserved bit
4	CCF4	PCA counter module 4 compare/capture flag bit clear (software write 0 to clear, write 1 is invalid), the read value is 1
3	CCF3	PCA counter module 3 compare/capture flag bit clear (software write 0 to clear, write 1 is invalid), the read value is 1
2	CCF2	PCA counter module 2 compare/capture flag bit clear (software write 0 to clear, write 1 is invalid), the read value is 1
1	CCF1	PCA counter module 1 compare/capture flag bit clear (software write 0 to clear, write 1 is invalid), the read value is 1
0	CCF0	PCA counter module 0 compare/capture flag bit clear (software write 0 to clear, write 1 is invalid), the read value is 1

Page 254**11.4.5 Compare capture mode register (PCA_CCAPM0~4)**

Offset address

CCAPM0: 0x010; CCAPM1: 0x014; CCAPM2: 0x018;

CCAPM3: 0x01C; CCAPM4: 0x020;

Reset value: 0x0000 0000

31:8	7	6	5	4	3	2	1	0
Reserved		ECOM	CAPP	CAPN	MAT	TOG	PWM	CCIE
		R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit symbol describe

31:7 Reserved reserved bit

6 ECOM Allow comparator function control bit

1: Enable the comparator function; 0: Disable the stronger function;

When PCA is used for software counter, high-speed output, PWM mode, WDT mode, ECOM must be set

Writing to the CCAMPPh or CCAMPx register will automatically set ECOM; writing to the CCAMPLx register will

Automatically clear the ECOM bit

5 CAPP Positive edge capture control bit

1: Allow rising edge capture; 0: Disable rising edge capture

4 CAPN Negative edge capture control bit

1: Allow falling edge capture; 0: Disable falling edge capture

3 MAT Allow matching control bits

1: Once the PCA count value matches the value of the compare/capture register of the module, the CCON register will be set

Break flag CCFx (x=0-4)

0: prohibit matching function

2 TOG Flip control bit

1: Working in PCA high-speed output mode, the value of the PCA counter is equal to the value of the module's compare/capture register

Once matched, the CCPx pin is flipped

0: disable rollover function

1 PWM Pulse width modulation control bit

1: Allow CCPx pin to be used as PWM output

0: Disable PWM pulse width modulation function

Only CCAPMx [6: 0] = 100_0010 time, the PWM function is effective

0	CCIE	PCA enable interrupt 1: Enable compare/capture interrupt 0: PCA compare/capture function interrupt is disabled
---	------	--

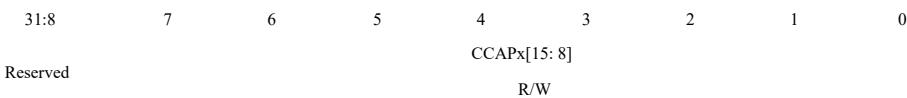
Page 255**11.4.6 Compare the upper 8 bits of the capture data register (**PCA_CCAP0~4H**)**

Offset address

CCAP0H: 0x024; CCAP1H: 0x02C; CCAP2H: 0x034;

CCAP3H: 0x03C; CCAP4H: 0x044;

Reset value: 0x0000 0000



Bit symbol describe

31:8 Reserved Reserved bit

7:0 CCAPx[15:8] compare/capture mode high 8-bit register
 When the PCA mode is used in the compare/capture mode, it is used to save the upper 8 bits of the 16-bit capture count value; write
 The CCAPxH register will automatically set the ECOM bit in the CCAPMx register.
 When the PCA mode is used in the PWM mode, it is used to control the output duty cycle and load the register.
 When the lower 8 bits overflow, the load register will be automatically updated to the PWM compare register

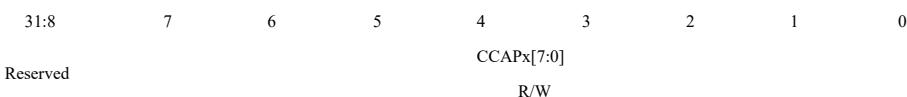
11.4.7 Compare the lower 8 bits of the capture data register (**PCA_CCAP0~4L)**

Offset address

CCAP0L: 0x028; CCAP1L: 0x030; CCAP2L: 0x038;

CCAP3L: 0x040; CCAP4L: 0x048;

Reset value: 0x0000 0000



Bit symbol describe

31:8 Reserved Reserved bit

7:0 CCAPx[7:0] compare/capture mode lower 8-bit register
 When PCA mode is used in compare/capture mode, it is used to save the lower 8 bits of the 16-bit capture count value; write
 The CCAPxH register will automatically clear the ECOM bit in the CCAPMx register.
 When the PCA mode is used in the PWM mode, it is used to control the output duty ratio compare register.
 Mode, the value of the lower 8 bits of the counter is less than the value of CCAPx[7:0] PWM output low level, otherwise
 PWM output high level.

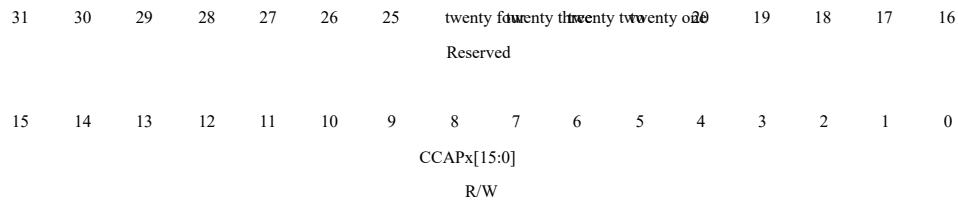
Page 256**11.4.8 Compare and capture 16-bit registers (PCA_CCAP0~4)**

Offset address

CCAP0: 0x050; CCAP1: 0x054; CCAP2: 0x058;

CCAP3: 0x05C; CCAP4: 0x060;

Reset value: 0x0000 0000

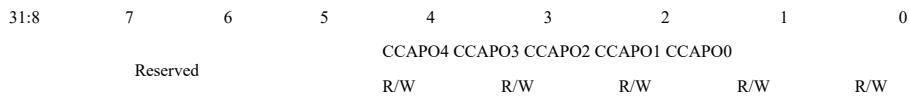


Bit	symbol	describe
31:16	Reserved	Reserved bit
15:0	CCAPx	<p>Compare/capture mode 16-bit register</p> <p>When PCA mode is used in compare/capture mode, it is used to save 16-bit capture count value; write CCAPx register</p> <p>The register will set the ECOM bit in the CCAPMx register.</p> <p>Writing the CCAPX register is equivalent to writing the two 8-bit registers CCAPxL and CCAPxH. In comparison/</p> <p>This register can be directly read and written in the capture mode. In the PWM mode, use CCAPxL and CCAPxH register</p>

11.4.9 Compare high-speed output flag register (PCA_CCAPO)

Offset address: 0x04C

Reset value: 0x0000 0000



Bit	symbol	describe
31:5	Reserved	Reserved bit
4	CCAPO4	Compare the output value of module 4
3	CCAPO3	Compare the output value of module 3
2	CCAPO2	Compare the output value of module 2
1	CCAPO1	Compare the output value of module 1
0	CCAPO0	Compare the output value of module 0

Page 257**12 advanced timer (TIM4/5/6)**

12.1 Introduction to Advanced Timer

The advanced timer is a Timer4/5/6 that contains three timers. Timer4/5/6 high-performance counter with the same function,

It can be used to count to generate different forms of clock waveforms. A timer can generate a complementary pair of PWM or independent

Two independent PWM outputs can capture external input for pulse width or period measurement.

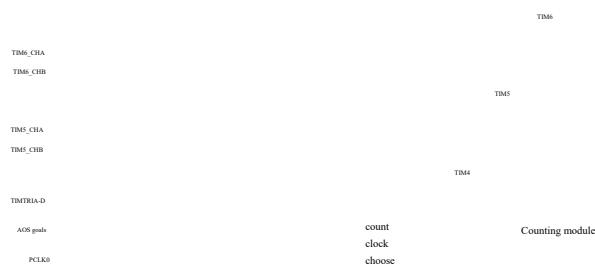
The basic functions and characteristics of advanced timers are shown in the table.

Wave mode	Sawtooth wave, triangle wave <ul style="list-style-type: none"> • Counting direction of increasing and decreasing • Software synchronization • Hardware synchronization
basic skills	<ul style="list-style-type: none"> • Cache function • Quadrature encoding count • Universal PWM output • Brake protection • AOS related actions Count compare match interrupt
Type of interrupt	Count cycle match interrupt Dead time error interrupt

Table 12-1 Basic features of Advanced Timer

Port name	direction	Function
TIMx_CHA	input	Quadrature encoding count clock input port or capture input port or comparison output
TIMx_CHB	Output	Outgoing port (x=4~6)
TIMTRIA		2) Input port for hardware start, stop and reset conditions
TIMTRIB		Hardware counting clock input port or capture input port
TIMTRIC	enter	Hardware start, stop, clear condition input port
TIMTRID		

Table 12-2 Advanced Timer port list



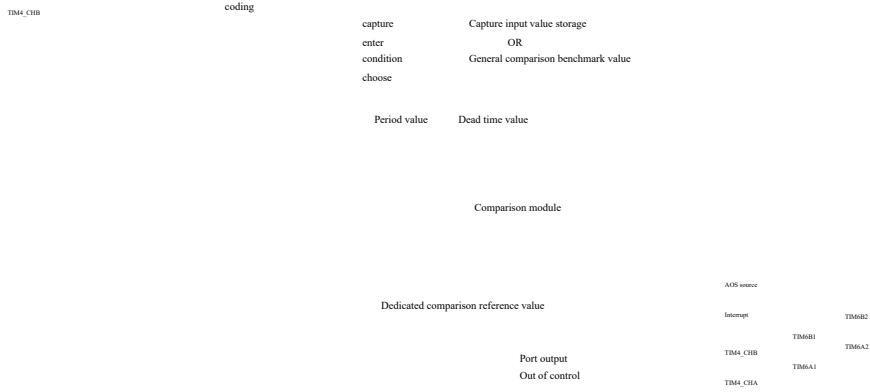


Figure 12-1 Block diagram of Advanced Timer

12.2 Advanced Timer function description

12.2.1 Basic actions

12.2.1.1 Basic Waveform Mode

Timer4/5/6 has 2 basic counting waveform modes, sawtooth wave mode and triangle wave mode. Wave mode is not

The same internal counting action is si

le wave mode is divided into triangle wave A mode and triangle wave B mode. Sawtooth w

The basic waveforms of triangle and ti

wn in Figure [12-2 and](#) Figure 12-3. The difference between triangle wave A mode and trian

There is a difference between the buffer transmission. In the triangular wave A mode, only one buffer transmission (valley point) occurs in a cycle, whi

The mode has two buffer transfers (peak and valley points) in one cycle.

CNTER.CNT[15:0]

FFFFH

0000H

GCONR.START

Figure 12-2 Sawtooth waveform (increasing counting)

CNTER.CNT[15:0]

FFFFH

0000H

T

GCONR.START

Figure 12-3 Triangle wave waveform

Page 260

12.2.1.2 Comparison output

Timer4/5/6 A timer has 2 comparison output ports (CHxA, CHxB), which can be

When the value compares and matches, the specified level is output. The GCMAR and GCMBR registers correspond to CHxA and CHxB respectively

Counting comparison base value. When the count value of the counter is equal to GCMAR, the CHxA port outputs the specified power

Level; When the count value of the counter is equal to GCMBR, the CHxB port outputs the specified level.

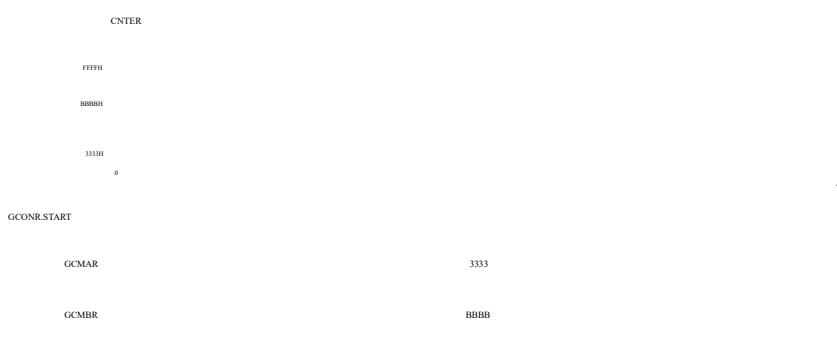
CHxA, CHxB port count start level, stop level, level at the time of counting comparison match, etc., can be used by the port

Control register (PCONR) PCONR.STACA, PCONR.STPCA, PCONR.STASTPSA,

PCONR.CMPCA[1:0], PCONR.PERCA[1:0] and PCONR.STACB, PCONR.STPCB,

PCONR.STASTPSB, PCONR.CMPCB[1:0], PCONR.PERCB[1:0] bit settings. [Figure 12-4](#) is

Operation example of comparison output.



CHB

PCONR.STAC[4]=1,PCONR.STACB=0
 PCONR.CMPCA[10]=PCONR.CMPCB[10]=11
 PCONR.PERCA[10]=PCONR.PERCB[10]=10

Figure 12-4 Comparison output action

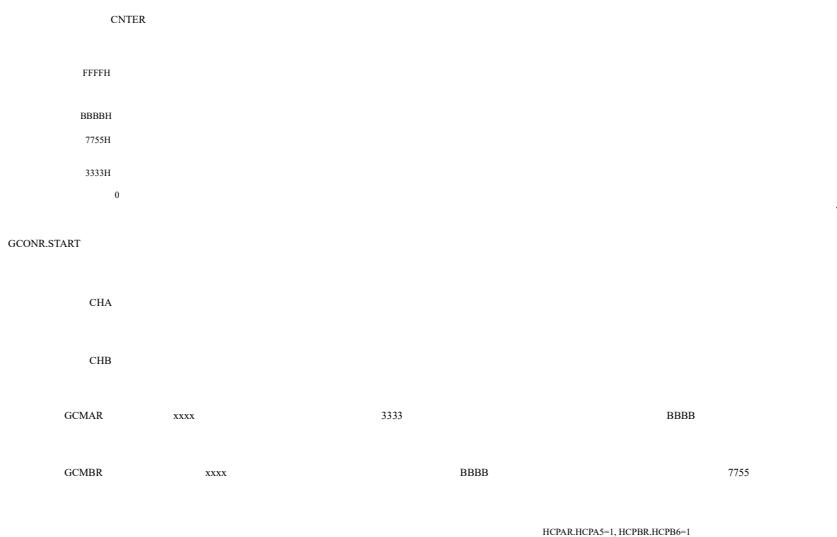
Page 261**12.2.1.3 Capture Input**

Figure 12-5 Capture input action

Timer4/5/6 have capture input function, with 2 sets of capture input registers (GCMAR, GCMBR), use

To save the captured count value. Set the PCONR.CAPCA of the port control register (PCONR),

When the PCONR.CAPCB bit is 1, the capture input function is valid. When the corresponding capture input conditions are set and the

When the condition is valid, the current count value is saved to the corresponding register (GCMAR, GCMBR).

The condition of each set of capture input can be AOS event trigger, TIMTRIA-TIMTRID input, CHxA or

CHxB input, etc., specific conditions can be selected through the hardware capture event selection register (HCPAR, HCPBR)

To set. Figure 12-5 is an example of capturing input actions.

Page 262**12.2.2 Clock source selection**

There are several options for the counting clock of Timer4/5/6:

- a. PCLK and PCLK divided by 2, 4, 8, 16, 64, 256, 1024 (GCONR.CKDIV[2:0] setting
Certainly)
- b. AOS event trigger input (HCUPR.HCUP [19:16] or HCDOR.HCDO [19:16] setting)
- c. Quadrature encoding input of CHxA and CHxB (HCUPR.HCUP[7:0] or HCDOR.HCDO [7:0]
set up)
- d. TIMTRIA-TIMTRID port input (HCUPR.HCUP [15:8] or HCDOR.HCDO [15:8]
set up)

It can be seen from the above description that b, c, d clocks are independent of each other, and can be set to be valid or invalid respectively, and when k When c, d clock, a clock is automatically invalid.

12.2.3 Counting direction

The counting direction of Timer4/5/6 can be changed by software. In different waveform modes, change the counting direction

The method is slightly different.

12.2.3.1 Sawtooth wave counting direction

In sawtooth wave mode, the counting direction can be set when the counter is counting or stopping.

When increasing counting, set GCONR.DIR=0 (decrement counting), then the counter will become decrement after counting to overflow

Counting mode; when counting down, set GCONR.DIR=1 (count up), then the counter will count to underflow

Then it becomes the increment counting mode.

When counting stops, set the GCONR.DIR bit. After the count starts until overflow or underflow, GCONR.DIR

The setting will be reflected in the count.

12.2.3.2 Triangular wave counting direction

In the triangle wave mode, the counting direction can only be set when the counter is stopped. Setting the counting direction in counting is invalid.

When counting stops, set the GCONR.DIR bit. After the count starts until overflow or underflow, GCONR.DIR

The setting will be reflected in the count.

12.2.4 Digital filtering

The CHxA, CHxB, and TIMTRIA~D port inputs of Timer4/5/6 all have digital filtering functions. Can be set by

The relevant enable bit of the filter control register (FCONR) enables the filter function of the corresponding port. Standard time for filtering

The clock is also set by the filter control register (FCONR).

When the filtered sampling reference clock is sampled to the same level on the port 3 times, the level is regarded as a valid level and sent to

Inside the module; Levels less than 3 times consistent will be filtered out as external interference and will not be transmitted to the inside of the module

For example, as shown in Figure [12-6](#).

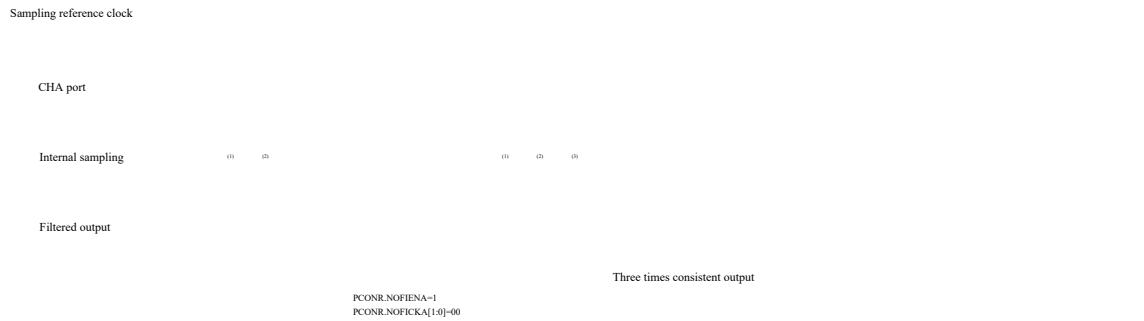


Figure 12-6 The filter function of the capture input port

The TIMTRIA~D ports are shared by a group of Timer4/5/6. The digital filter function of this group of ports is only available in Timer4 is implemented, and other timers, Timer5/6, have no effect on the digital filter function setting of this group of ports.

12.2.5 Software synchronization

12.2.5.1 Synchronous start of software

Timer4/5/6 can achieve the goal Timer4/5/6 by setting the relevant bits of the software synchronization start register (SSTAR)

The synchronization starts.

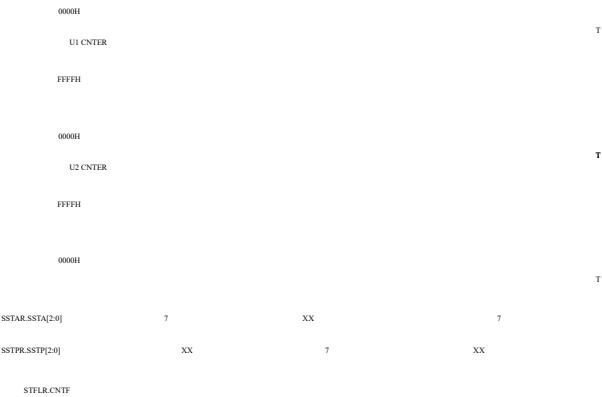


Figure 12-7 Software synchronization action

12.2.5.2 Software synchronization stops

Timer4/5/6 can achieve the goal Timer4/5/6 by setting the relevant bits of the software synchronous stop register (SSTPR)

Sync stopped.

12.2.5.3 Software Synchronous Clear

Timer4/5/6 can be set to the relevant bits of the software synchronous clear register (SCLRR) to achieve the target Timer4/5/6

The synchronization is cleared.

As shown in Figure [12-7](#), if you set Timer4's SSTAR.SSTA0=SSTAR.SSTA1=SSTAR.SSTA2, that is

The software can be started synchronously for Timer4/5/6.

Software synchronization action related registers (SSTAR, SSTPR, SCLRR) are a set of independent

The registers shared between each TIM, each bit of this group of registers is only valid when writing 1 and writing 0 is invalid. Reading

When the SSTAR register is read, the counter status of each timer will be read. When SSTPR or SCLRR is read, it will be

Read 0.

Page 267

12.2.6 Hardware synchronization

In addition to independently having 2 general-purpose input ports (CHxA, CHxB), each timer also has 4 external Universal input ports (TIMTRIA, TIMTRIB, TIMTRIC, TIMTRID) and 4 AOS targets, which can be Realize hardware synchronization actions between timers.

12.2.6.1 Hardware synchronization start

Each Timer4/5/6 can choose to start the counter by hardware, just select the timer with the same hardware start condition Synchronous start is realized when the start condition is valid. The specific hardware start condition is selected by the hardware start event selection register (HSTAR) setting to decide.

12.2.6.2 Hardware synchronization stop

Each Timer4/5/6 can choose to stop the counter by hardware, just select the timer with the same hardware stop condition A synchronous stop is realized when the stop condition is valid. The specific hardware stop condition is selected by the hardware stop event register (HSTPR) is determined by the setting.

12.2.6.3 Hardware Synchronous Clear

Each Timer4/5/6 can choose to clear the counter by hardware, just select the timer with the same hardware clearing condition Synchronous clearing is achieved when the clearing condition is valid. The specific hardware reset condition is selected by the hardware reset event selection register (HCLR) is determined by the setting.

12.2.6.4 Hardware synchronization capture input

Each Timer4/5/6 can choose to use hardware to realize the capture input function, and select those with the same capture input function condition.

The timer can realize the synchronous capture input when the condition of the capture input function is valid. Specific hardware capture input function.

The condition is determined by the setting of the hardware capture event selection register (HCPAR, HCPBR).

12.2.6.5 Hardware synchronization counting

Timer4/5/6 can choose to use hardware input as CLOCK for counting.

The timer can realize synchronous counting when the hardware counting CLOCK is valid. The specific hardware counting conditions are incremented Event selection register (HCUPR) and hardware decrement event selection register (HCDOR) are set to determine.

When the hardware synchronous counting function is selected, only the external input clock source is selected, which does not affect the start, stop, and

Page 268

Clear action. The start, stop, and reset of the counter also need to be set separately.

[Figure 12-8](#) shows an example of hardware synchronization operation of Timer4/5/6.



Figure 12-8 Hardware synchronization action

Page 269**12.2.7 Cache function**

Cache action means that by setting the cache control register (BCONR), at the time of cache transfer, select to occur

The next event:

- a. The value of the general period reference value buffer register (PERBR) is automatically transferred to the general period reference value register (PERAR)
- b. The value of the general comparison reference value buffer register (GCMCR, GCMDR) is automatically transferred to the general comparison reference value register (GCMAR, GCMBR) (when comparing output)
- c. The value of the general comparison reference value register (GCMAR, GCMBR) is automatically transferred to the general comparison reference value register (GCMCR, GCMDR) (when capturing input)

[Figure 12-9](#) shows the timing chart of the single-buffer mode of the general-purpose comparison reference value register during the comparison output. As you can see in the figure, changing the value of the general comparison reference value register (GCMAR) during the counting period can adjust the comparison output. The duty cycle can be adjusted by changing the value of the universal period reference value register (PERAR).

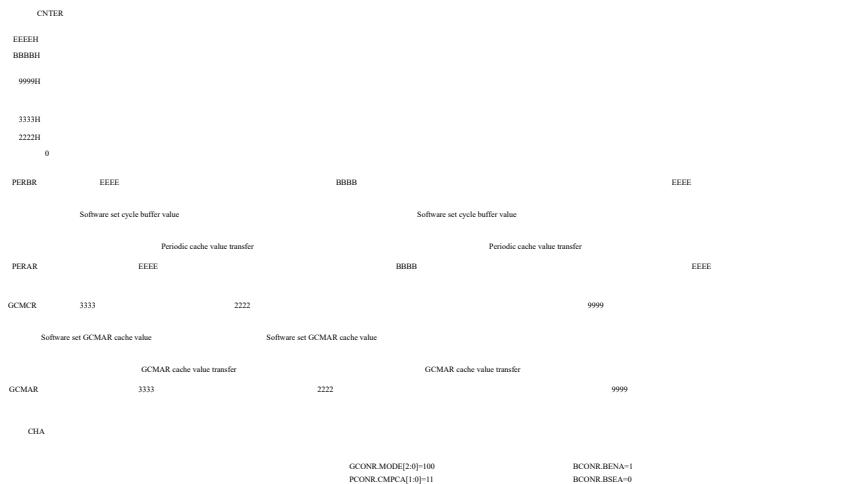


Figure 12-9 Comparison output timing of single buffer mode

Page 270**12.2.7.1 Cache transfer time point****12.2.7.2 Common cycle reference value buffer transmission time point**

When the cycle reference value buffer transmission time point is a sawtooth wave, the counting up overflow point or the down counting underflow point. Count valley points.

12.2.7.3 General comparison reference value buffer transmission time point

In sawtooth wave A mode, set BCONR.BENA=1 or BCONR.BNEB=1, the buffer action is valid. Cache The transfer occurs at the overflow point or the underflow point.

In triangle wave A mode, set BCONR.BENA=1 or BCONR.BNEB=1, the buffer action is valid. Cache The transmission occurs at the counting valley point.

In triangular wave B mode, set BCONR.BENA=1 or BCONR.BNEB=1, the buffer action is valid. Cache The transmission occurs at the counting valley point and the counting peak point.

12.2.7.4 Capture input value buffer transmission time point

The capture input action buffer transfer time point is when the capture input action.

12.2.7.5 Buffer transmission **during clearing action**

In sawtooth wave counting mode or hardware counting mode, if there is a clearing action during the normal comparison output action, The general cycle reference value, general comparison reference value, etc. will be cached once according to the corresponding cache action setting start Transmit.

Page 271

12.2.8 General PWM output

12.2.8.1 PWM spread spectrum output

In order to reduce the interference of PWM output to the outside, there is a spread spectrum configuration in the PWM output stage. Each PWM output The period will fine-tune the phase of the PWM output.

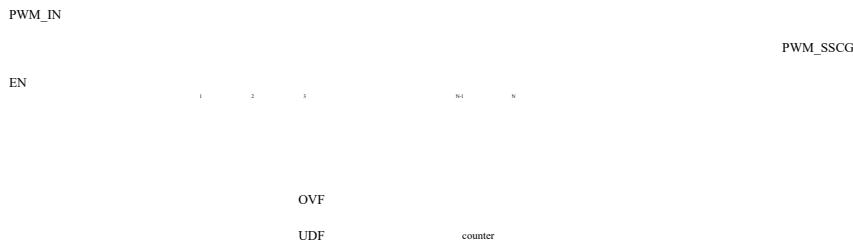


Figure 12-10 Schematic diagram of PWM spread spectrum output

12.2.8.2 Independent PWM output

The 2 ports CHxA and CHxB of each timer can output PWM waves independently. As shown in Figure 12-11 , the timer

The CHA port of Timer6 outputs PWM wave.



Figure 12-11 CHA output PWM wave

12.2.8.3 Complementary PWM output

The CHxA port and CHxB port can be combined to output complementary PWM waveforms in different modes.

12.2.8.3.1 Software setting GCMBR complementary PWM output

The software setting GCMBR complementary PWM output means that in sawtooth wave mode and triangle wave A mode, triangle wave B

In the mode, the value of the general comparison reference value register (GCMBR) used for the waveform output of the CHxB port is set by the re

The register is set directly and has no direct relationship with the value of the general comparison reference value register (GCMAR).

[Figure 12-12](#) is an example of software setting GCMBR complementary PWM wave output.

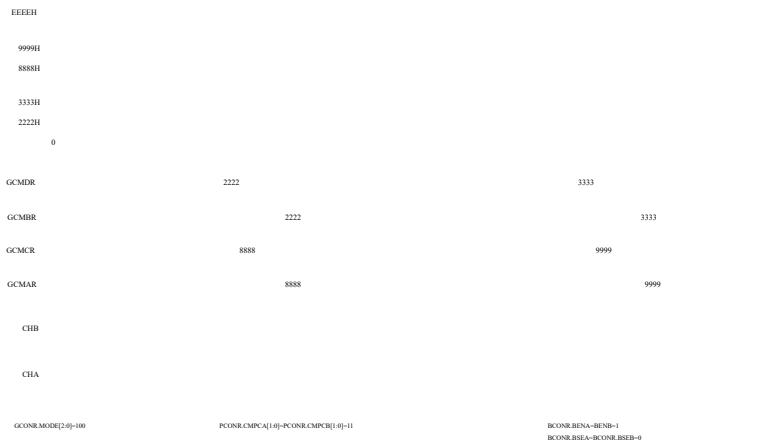


Figure 12-12 The software setting GCMBR complementary PWM wave output in triangle wave A mode

12.2.8.3.2 Hardware setting GCMBR complementary PWM output

Hardware setting GCMBR complementary PWM output means that in triangle wave A mode and triangle wave B mode, it is used for The value of the general comparison reference value register (GCMBR) of the waveform output of the CHxB port is determined by the general com The value of the register (GCMAR) and the dead time reference value register (DTUAR, DTDAR) are determined by calculation.

[Figure](#) 12-13 is an example of hardware setting GCMBR complementary PWM wave output.



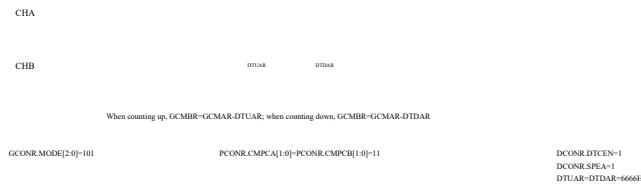


Figure 12-13 Hardware setting of GCMBR complementary PWM wave output in triangle wave B mode (symmetrical dead zone)

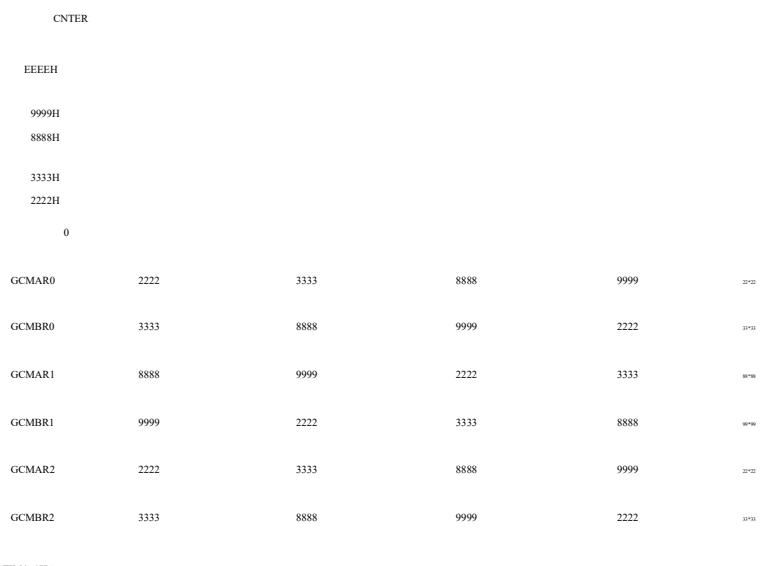
12.2.8.4 Multi-phase PWM output

The CHxA and CHxB ports of each timer can output 2-phase independent PWM waves or a set of complementary PWM

Multi-phase PWM wave output can be realized by combining multiple timers w ; of software and hardware.

As shown in Figure 12-14 , Timer4, Timer5, and Timer6 combine to output 6-pl own in Figure 12-15, Timer4,

The combination of Timer5 and Timer6 outputs 3-phase complementary PWM waves.



TIM4_CHB

TIM5_CHA

TIM5_CHB

TIM6_CHA

TIM6_CHB

GCONRx.MODE[2:0]=000
 GCONRx.CMPCA[1:0]=PCONRx.CMPCB[1:0]=01
 GCONRx.PERCA[1:0]=PCONRx.PERCB[1:0]=00
 (X=0, 1, 2)

Figure 12-14 6-phase PWM wave

Page 275

CNTER

EEEEEH

9999H
8888H3333H
2222H

0

T

GCMAR0	9999	8888	9999
GCMBR0	5555	4444	5555
GCMAR1	8888	9999	8888
GCMBR1	6666	7777	6666
GCMAR2	3333	2222	3333
GCMBR2	2222	1111	2222

TIM4_CHA

DTUAR0 DTDAR0

TIM4_CHB

TIM5_CHA

DTUAR1 DTDAR1

TIM5_CHB

TIM6_CHA

DTUAR2 DTDAR2

TIM6_CHB

When counting up, GCMBR=GCMAR-DTUAR; when counting down, GCMBR=GCMAR-DTDAR

GCONRx.MODE[2:0]=100
(X=0, 1, 2)GCONRx.CMPCA[1:0]=PCONRx.CMPCB[1:0]=11
(X=0, 1, 2)

DCONRx.DTCEN=1
 DCONRx.SPEA=1
 DTUAR0-DTDAR0=4444H
 DTUAR1-DTDAR1=2222H
 DTUAR2-DTDAR2=1111H
 (X=0, 1, 2)

Figure 12-15 Three-phase complementary PWM wave output with dead time in triangle wave A mode

Page 276

12.2.9 Quadrature Encoding Count

Regard CHxA input as AIN input, CHxB input as BIN input, or any one of TIMTIA-D

Two inputs are regarded as ZIN inputs, and Advanced Timer can realize the quadrature encoding counting of three inputs.

The position counting mode can be realized by the AIN and BIN actions of one timer; the AIN, BIN and BIN of two timers

ZIN combined action can realize revolution counting mode, one timer is used for position counting, and one timer is used for public revolution.

Turn count.

In revolution counting mode, every two timers are combined (timers 4 and 5 are combined, and timer 4 is used as the position counting unit

Yuan, timer 5 is used as revolution counting unit) to realize position counting and revolution counting respectively.

The counting conditions of AIN and BIN are set by the hardware decrement event selection register (HCUPR) and hardware decrement event.

The orthogonal relationship between CHxA and CHxB in the file selection register (HCDOR) is realized; the input action of ZIN is through

Set the hardware reset event selection register (HCLRR) of the position unit to realize the position counter of the position counting unit

Clear, realize the revolution counting unit by setting the hardware incremental event selection register (HCUPR) of the revolution unit

The revolution counter counts.

12.2.9.1 Position counting mode

Quadrature encoding position mode refers to the basic counting function and phase difference counting function according to the input of AIN and BIN

And direction counting function.

12.2.9.1.1 Basic counting

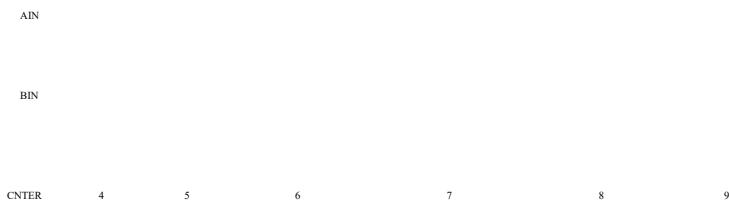


Figure 12-16 Basic counting action in position mode

By setting the HCUPR and HCDOR registers, various methods of phase difference counting can be realized flexibly.

Page 277**12.2.9.1.2 Phase difference counting**

Phase difference counting refers to counting according to the phase difference between two inputs.

There are 1 times counting, 2 times counting, 4 times counting, etc

IN and BIN. Depending on the setting, it can be implemented.

Figure 12-19 below.



Figure 12-17 Phase difference counting action setting in position mode (1 times)

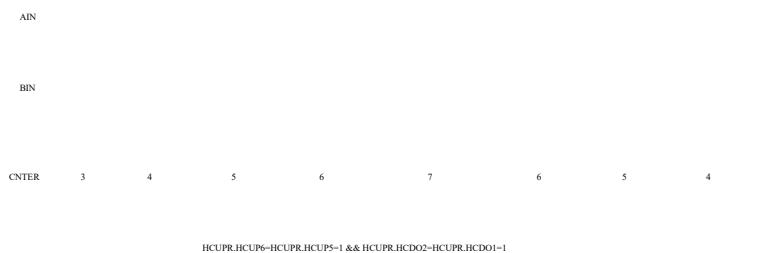


Figure 12-18 Phase difference counting action setting in position mode (2 times)

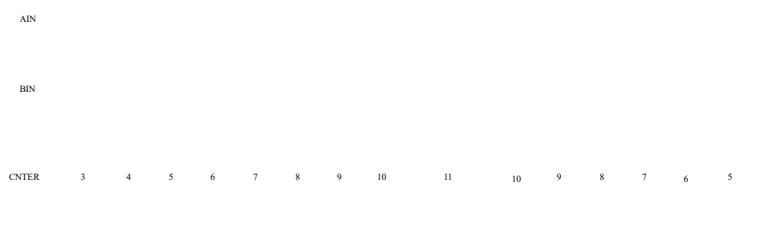


Figure 12-19 Phase difference counting action setting in position mode (4 times)

Page 278

12.2.9.1.3 Direction count

Direction counting refers to setting the input state of AIN as direction control, and counting the input of BIN as a clock.

As shown in Figure 12-20 .



Figure 12-20 Direction counting action in position mode

12.2.9.2 Revolution mode

Orthogonal coding revolution mode means that on the basis of AIN and BIN counting, the input event of ZIN is added to realize Judgment of the number of revolutions, etc. According to the counting method of the revolution counter in the revolution mode, it can realize the Z-phase Position counter output counting function and Z-phase counting and position counter output mixed counting function. I.e. use two Advanced Timer realizes this function.

12.2.9.2.1 Z phase count

Z-phase counting means that according to the input of ZIN, the revolution counting unit counts and the position counting unit is cleared at the same time. The counting action.

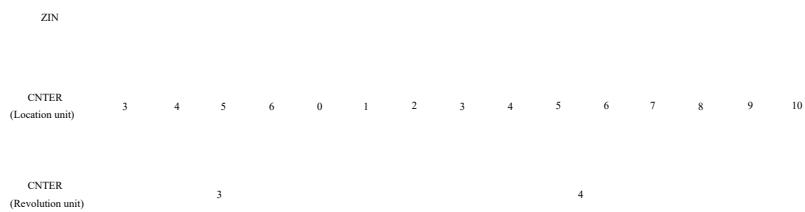


Figure 12-21 Phase Z counting action in revolution mode

12.2.9.2.2 Position overflow count

The position overflow counting means that when the position counting unit overflows or underflows, an overflow event is generated, thereby triggering the counter of the sending revolution counting unit performs a count (in this counting mode, the input of ZIN does not perform revolution counting). The counting action of the counting unit and the clearing action of the position counting unit). The overflow event of the position counting unit realizes the counting of revolution counting unit through the linkage strobe of the AOS module.

The current position overflows the count. The hardware increment (decrement) event selection register (HCUPR or HCDOR) Increasing (decreasing) event selects 1 bit in Bit16:it19, and the AOS module setting corresponds to

The event source of the increment (decrement) event is the counting overflow event of the position counting unit. For details, please refer to the AOS Festival. As shown in Figure [12-22](#).

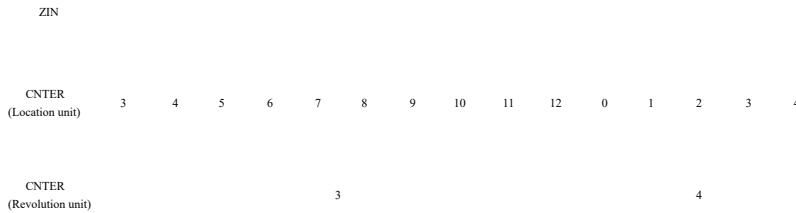


Figure 12-22 Position counter output counting action in revolution mode

12.2.9.2.3 Mixed counting

Mixed counting refers to the counting action that combines the two counting methods of Z-phase counting and position overflow counting.

The realization method is also a combination of the above two counting methods. As shown in Figure [12-23](#).

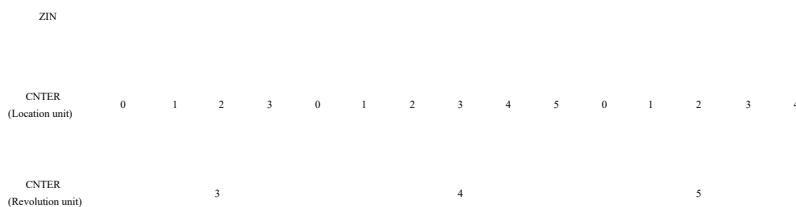


Figure 12-23 Mixed counting action of Z-phase counting and position counter output in revolution mode

12.2.9.2.4 Z- phase action shield

When the Z-phase counting function or the mixed counting function in the revolution counting mode is used, the overflow of the position counter can In a few cycles after the point or underflow point (GCONR.ZMSK[0:1] setting), the effective input of ZIN is shielded,

The counting of the revolution counting unit and the clearing of the position counting unit are not performed.

When the GCONR.ZMSKPOS of the general control register (GCONR) of the position counter unit is 1, the position counter

The Z-phase shielding function of the digital unit is enabled, and the number of cycles of the Z-phase shielding is set by GCONR.ZMSK; revolution

When the GCONR.ZMSKREV of the general control register (GCONR) of the unit is 1, the revolution counting unit's

Z phase shielding function is enabled.

[Fig. 12-24](#) is the 4 count cycles after the position counting unit count overflows when the revolving count mode is mixed counting

When there is a ZIN phase input, the action of the ZIN phase input is invalid, that is, the revolution counting unit does not count, and the position cou

The element is not cleared; the ZIN phase input coming later works normally.

ZIN

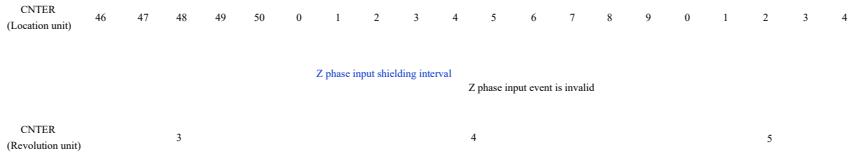


Figure 12-24 Revolution counting mode-mixed counting Z phase shielding action example 1

[Figure 12-25](#) is the third cycle after the position counting unit count overflows when the revolution counting mode is mixed counting.

The counting direction changes, and the set masking period of 4 cycles becomes invalid at this time (the actual ZIN phase masking power can be maintained for 3 cycles) and starts counting down. After the counting underflow occurs in the position counting unit, the ZIN phase is shielded. The function is turned on again and becomes invalid after maintaining for 4 cycles. During the ZIN phase shielding period, the input function of the \bar{Z} terminal is invalid, that is, the revolution counting unit does not count, and the position counting unit is not cleared; the ZIN phase input afterwards is normal action.

HC32L110 Series User Manual Rev2.31

Page 280 of 527

Page 281

ZIN



Figure 12.25: Results of a simulation showing the effect of Zeta on different time scales.

HC32L110 Series User Manual Rev2.31

Page 281 of 527

Page 282

12.2.10 Periodic interval response

Timer4/5/6 general comparison reference value register (GCMAR~GCMDR), which can be divided when the count comparison matches

Do not generate a dedicated valid request signal and send it to the AOS module for associated actions with other modules.

The request signal can generate a valid request signal every few cycles. Send by setting the effective period

The VPERR.PCNTS bit of the register (VPERR) is used to specify how many cycles the request signal is valid.

Even if the count value is equal to the value of the comparison reference value register GCMAR or GCMBR during its cycle, it will not

A valid request signal is output. Figure 12-26 shows an example of the operation of the periodic interval valid request signal.

Figure 12-26 Periodic interval valid request signal action

Page 283

12.2.11 Protection mechanism

Advanced Timer can protect and control the output status of the port.

Advanced Timer has 4 common ports to output invalid event interfaces, these 4 interfaces are connected to the brake control module

4 groups of braking events output by the block. The abnormal condition events strobed on each interface can be set from the brake control. When the:

When an abnormal condition is detected on the interface, the general PWM output can be controlled.

During the normal output period of the port, if a brake event from the brake control is detected, the output state of the port can be changed

It is a preset state. General PWM output port in the event of abnormal brake control, port status

It can be changed to output high impedance state, output low level or output high level (PCONR.DISVALA,

PCONR.DISVALB setting decision).

For example, if PCONR.DISSELA[1:0]=01&PCONR.DISVALA=01 is set,

During the normal output of the port, if a brake event occurs on the output invalid condition 1, the output on the CHxA port becomes high.

Resistance state.

Page 284**12.2.12 Interrupt description**

Timer4/5/6 each contains 3 types of 9 interrupts in total. They are 4 general-purpose counting compare match interrupts (including 2 Capture input interrupt), 2 count cycle match interrupt, 1 dead time error interrupt.

12.2.12.1 Counting compare match interrupt

There are a total of 4 general comparison reference value registers (GCMAR-GCMDR), which can be compared with the count value to generate a raw matching effective signal. When the count comparison matches, the status flag register (STFLR)

The STFLR.CMAF~STFLR.CMDF bits will be set to 1 respectively. If the interrupt control register is set at this time (ICONR) the corresponding bit in ICONR.INTENA~ICONR.INTEND is 1 to enable interrupt, then the corresponding

The interrupt request will also be triggered.

When the capture input valid condition selected by the hardware capture event selection register (HCPAR, HCPBR) is generated,

The capture input action occurs. At this time, if you set the ICONR.INTENA of the interrupt control register (ICONR) or

If the ICONR.INTENB bit is 1 to enable the interrupt, the corresponding interrupt request is triggered.

12.2.12.2 Count period match interrupt

Sawtooth wave counts up to the overflow point, sawtooth wave counts down to the underflow point, triangle wave counts to the valley point or triangle

When counting to the peak point, the STFLR.OVFF or STFLR.UDFF bit of the status flag register (STFLR) will be

Set to 1. At this time, if you set the ICONR.INTENOVF bit of the interrupt control register (ICONR) and

The ICONR.INTENUDF bit enables interrupts, and the counting cycle match interrupt can be triggered at the corresponding point in time.

12.2.12.3 Dead time error interrupt

Load the value of the dead time reference value register (DTUAR, DTDAR) to the general comparison reference value register

(GCMBR), if the cycle limit is exceeded, a dead time error will occur, and the status flag register (STFLR)

The STFLR.DTEF bit will be set to 1. At this time, if the interrupt control register (ICONR) is set

If the ICONR.INTENDE bit enables interrupts, the dead time error interrupt will be triggered at that moment.

Page 285

12.2.13 Brake protection

When invalid conditions 0~3 can be set, configure PCONR.DISVALA, PCONR.DISVALB. Invalid conditions are

The hardware will automatically change the port state to the preset state (high level, low level, high impedance state, and maintain normal output) wh

12.2.13.1 Port brake and software brake

The port is controlled by polarity selection. After it is effectively enabled, it is digitally filtered and synchronized to generate a port brake flag;

The port brake flag is used as an invalid condition of Advanced Timer3. The port brake flag needs to be cleared by software.

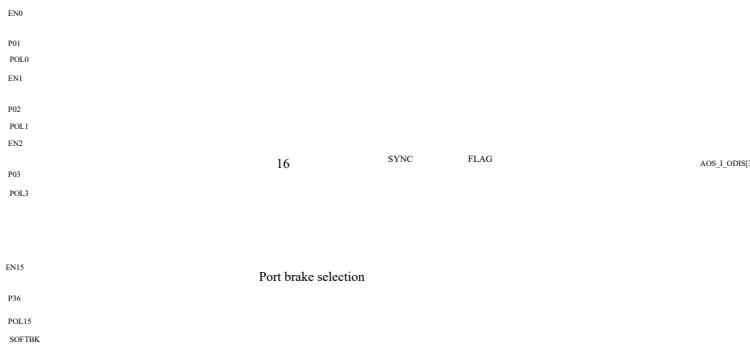


Figure 12-27 Schematic diagram of port brake and software brake

12.2.13.2 Automatic brake in low power consumption mode

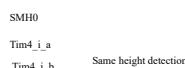
The system enters the low power consumption mode, and the PWM will not work normally after the clock is stopped. Low power mode as

Invalid condition 2 of Advanced Timer controls PWM brake.

12.2.13.3 The output level is the same as the high and the same as the low brake

The output level is monitored by level, and after it is effectively enabled, after synchronization, the same high and the same low brake flag will be gei

Flag as invalid condition 1 of Advanced Timer. The same high and same low brake mark needs to be cleared by software.



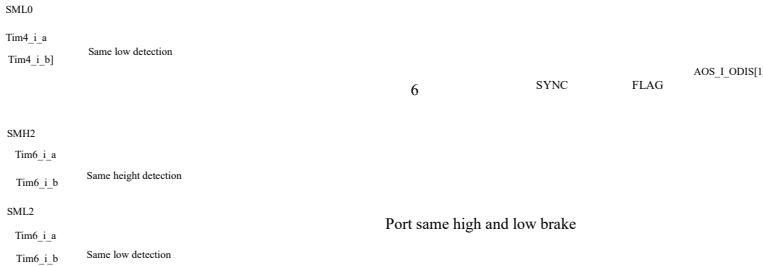


Figure 12-28 Schematic diagram of output with the same high and low brake

12.2.13.4 VC brake

VC1, VC2 interrupt flags are enabled as the invalid condition 0 of Advanced Timer.



Figure 12-29 Schematic diagram of VC brake control

12.2.14 Internal interconnection

12.2.14.1 Interrupt trigger output

Because one interrupt of Timer4/5/6 contains multiple interrupt sources. Control the interrupts that trigger the ADC and control the AOS. The signal can be individually controlled to select different sources, you can select overflow, underflow, 4 comparison matches, a total of 6. Any interrupt source of TIMx interrupt source is used as the trigger condition.



CMDE				
INT_OVF				
OVFE				
INT_UDF			TIMx interrupt selection	
UDFE				

Figure 12-30 Timer4/5/6 interrupt selection

12.2.14.2 AOS trigger

AOS is an internal signal of the system, which can trigger the on of the Advanced Timer counter after selecting and controlling.

Start, stop, clear, add 1, subtract 1 and other functions. Advanced Timer has 4 AOS triggers, each trigger

The interrupt source of different modules can be selected. The selected signal generates a single pulse trigger and is input to the Advanced Timer,

Control the start, stop, and clear of the counter of the Advanced Timer.

Timer4/5/6 uses registers internally to select different AOS_I_TRIGGER as its own trigger signal. If you can

The HSTAR register can be used to trigger the hardware start of the corresponding timer using an interrupt.

	AOS_i_trig0	AOS_i_trig1	AOS_i_trig2	AOS_i_trig3
Select control signal ITRIG.IAOS0S ITRIG.IAOS1S ITRIG.IAOS2S	ITRIG.IAOS3S			
0000	TIM0_INT	TIM0_INT	TIM0_INT	TIM0_INT
0001	TIM1_INT	TIM1_INT	TIM1_INT	TIM1_INT
0010	TIM2_INT	TIM2_INT	TIM2_INT	TIM2_INT

Page 288

	LPTIMER_INT	LPTIMER_INT	LPTIMER_INT	LPTIMER_INT
0100	TIM4_INTS	TIM4_INTS	TIM4_INTS	TIM4_INTS
0101	TIM5_INTS	TIM5_INTS	TIM5_INTS	TIM5_INTS
0110	TIM6_INTS	TIM6_INTS	TIM6_INTS	TIM6_INTS
0111	UART0_INT	UART0_INT	UART0_INT	UART0_INT
1000	UART1_INT	UART1_INT	UART1_INT	UART1_INT
1001	LPUART_INT	LPUART_INT	LPUART_INT	LPUART_INT
1010	VC1_INT	VC1_INT	VC1_INT	VC1_INT
1011	VC2_INT	VC2_INT	VC2_INT	VC2_INT
1100	RTC_INT	RTC_INT	RTC_INT	RTC_INT
1101	PCA_INT	PCA_INT	PCA_INT	PCA_INT
1110	SPI_INT	SPI_INT	SPI_INT	SPI_INT
1111	ADC_INT	ADC_INT	ADC_INT	ADC_INT

Table 12-3 AOS source selection

12.2.14.3 Port trigger TRIGA-TRIGD

Port triggering can control the hardware start, stop, clear, capture, counter up and down of Advanced Timer

Digital filtering function is optional, and the port can be configured as any port of the chip.

Independent selection of control signal	TRIGA	TRIGB	TRIGC	TRIGD
control				
0000	P01	P01	P01	P01
0001	P02	P02	P02	P02

0010	P03	P03	P03	P03
0011	P15	P15	P15	P15
0100	P14	P14	P14	P14
0101	P23	P23	P23	P23
0110	P24	P24	P24	P24
0111	P25	P25	P25	P25
1000	P26	P26	P26	P26
1001	P27	P27	P27	P27
1010	P31	P31	P31	P31
1011	P32	P32	P32	P32
1100	P33	P33	P33	P33
1101	P34	P34	P34	P34
1110	P35	P35	P35	P35
1111	P36	P36	P36	P36

Table 12-4 Port trigger selection

12.2.14.4 Compare output VC and Advanced Timer interconnection

VC can be interconnected to the capture input terminal of Advanced Timer, which can capture the edge of VC output.

Get

12.2.14.5 UART and Advanced Timer interconnection

UARTx_RX / LPUART_RX can be internally interconnected to BaseTimer, LPTimer, PCA and

Advanced Timer. The automatic identification of baud rate can be realized through software.

The UART selection control register is in the port control register GPIO_CTRL3, and the VC output control register is in the port control register VC control module.

Page 290**12.3 Register description**

CH0 base address 0x40003000

CH1 base address 0x40003400

CH2 base address 0x40003800

register	Offset address	describe
TIMx_CNTER	0x000	General Counting Reference Value Register
TIMx_PERAR	0x004	Universal Period Reference Value Register
TIMx_PERBR	0x008	General Cycle Reference Value Cache Register
TIMx_GCMAR	0x010	General compare A reference value register
TIMx_GCMBR	0x014	General compare B reference value register
TIMx_GCMCR	0x018	General compare C reference value register
TIMx_GCMDR	0x01C	General comparison D reference value register
TIMx_DTUAR	0x040	Dead time reference value register
TIMx_DTDAR	0x044	Dead time reference value register
TIMx_GCONR	0x050	General control register
TIMx_ICONR	0x054	Interrupt control register
TIMx_PCONR	0x058	Port control register
TIMx_BCONR	0x05C	Cache control register
TIMx_DCONR	0x060	Dead zone control register
TIMx_FCONR	0x068	Filter control register
TIMx_VPERR	0x06C	Valid period register
TIMx_STFLR	0x070	Status flag register
TIMx_HSTAR	0x074	Hardware start event selection register
TIMx_HSTPR	0x078	Hardware stop event selection register
TIMx_HCELR	0x07C	Hardware clear event selection register
TIMx_HCPAR	0x080	Hardware capture event selection register
TIMx_HCPBR	0x084	Hardware capture event selection register
TIMx_HCUPR	0x088	Hardware decrement event selection register
TIMx_HCDOR	0x08C	Hardware decrement event selection register
TIMx_IFR	0x100	Interrupt flag register
TIMx_ICLR	0x104	Interrupt clear register
TIMx_CR	0x108	Spread spectrum and interrupt trigger selection register
TIMx_AOSSR	0x110	AOS selection register, shared by three channels
TIMx_AOSCL	0x114	AOS brake flag clear register, shared by three channels
TIMx_PTBKSL	0x118	Port brake control register, shared by three channels
TIMx_TTRIG	0x11C	Port trigger control register, shared by three channels

Page 291

TIMx_PTAKP	0x124	Port brake polarity control register, shared by three channels
TIMx_SSTAR	0x3F4	Software synchronization start register
TIMx_SSTPR	0x3F8	Software synchronization stop register
TIMx_SCLRR	0x3FC	Software synchronization clear register

Table 12-5 Advanced Timer register list

Page 292

12.3.1 General Counting Reference Value Register (TIMx_CNTER)

Address offset: 0x000

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															

R/W

Bit	symbol	Function description
31:16	Reserved	-
15:0	CNT[15:0]	Current counter value

12.3.2 General Period Reference Value Register (TIMx_PERAR)

Address offset: 0x004

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PERA[15:0]															

R/W

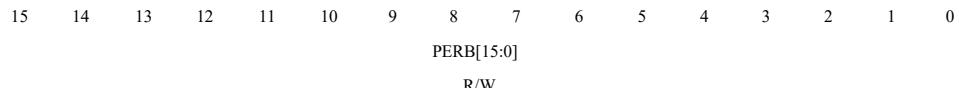
Bit	symbol	Function description
31:16	Reserved	-
15:0	PERA[15:0]	Counting cycle value, set the counting cycle value for each round of counting

12.3.3 General Period Buffer Register (TIMx_PERBR)

Address offset: 0x008

Reset value: 0x0000 FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

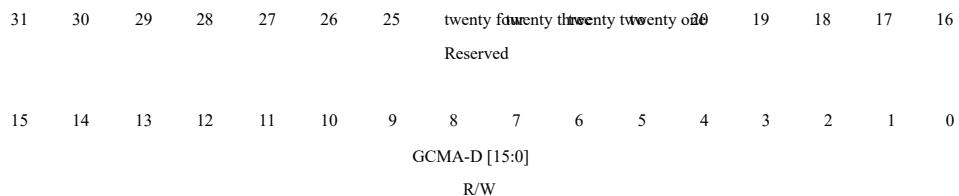


Bit	symbol	Function description
31:16	Reserved	-
15:0	PERB[15:0]	Cache count period value, cache value of count period

12.3.4 General comparison reference value register (**TIMx_GCMAR-GCMDR**)

Address offset: 0x0010, 0x0014, 0x0018, 0x001C

Reset value: 0x0000 FFFF

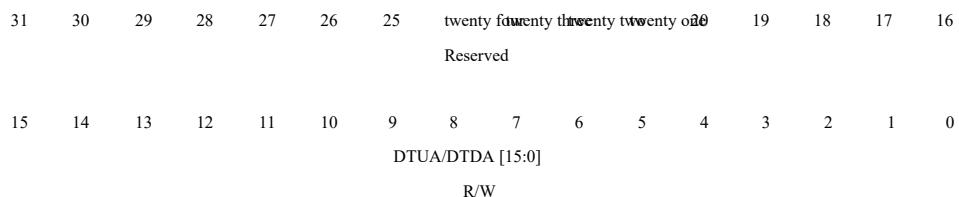


Bit	symbol	Function description
31:16	Reserved	-
15:0	GCMA-D [15:0]	Counting comparison reference value, comparison reference value setting, the matching signal is valid when the count value is equal

12.3.5 Dead-time reference value register (**TIMx_DTUAR- DTDAR**)

Address offset: 0x040, 0x044

Reset value: 0x0000 FFFF



Bit	symbol	Function description
31:16	Reserved	-

15:0 DTUA/DA [15:0] Dead time value, dead time setting value

Page 295**12.3.6 General Control Register (TIMx_GCONR)**

Address offset: 0x050

Reset value: 0x00000100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
							DIR	Res.	CKDIV		MODE		START		
							R/W		R/W		R/W		R/W		

Bit	mark	Function description
31:20	Reserved	-
19:18	ZMSK	Z phase input shielding cycles Quadrature encoding Z phase input is masked counting period value 00: Z phase input shielding function is invalid 01: The Z-phase input in the 4 counting cycles after the position count overflows or underflows is masked 10: The Z-phase input in 8 count cycles after the position count overflows or underflows is masked 11: The Z-phase input in 16 count cycles after the position count overflows or underflows is masked
17	ZMSKPOS	Z phase input position counter selection 0: The timer is used as a position counter when phase Z is input, and the position counter is cleared during the mask period

19:18	ZMSK	Z phase input shielding cycles Quadrature encoding Z phase input is masked counting period value 00: Z phase input shielding function is invalid 01: The Z-phase input in the 4 counting cycles after the position count overflows or underflows is masked 10: The Z-phase input in 8 count cycles after the position count overflows or underflows is masked 11: The Z-phase input in 16 count cycles after the position count overflows or underflows is masked
17	ZMSKPOS	Z phase input position counter selection 0: The timer is used as a position counter when phase Z is input, and the position counter is cleared during the mask period

Normal action

1: The timer is used as a position counter when phase Z is input, and the position counter is cleared during the mask period
 Hidden

16 ZMSKREV Z phase input revolution counter selection

0: The timer is used as a revolution counter when phase Z is input, and the revolution counter counts during the mask period
 Normal action

1: The timer is used as a revolution counter when phase Z is input, and the revolution counter counts during the mask period
 Hidden

15:9	Reserved	-
8	DIR	Counting direction 0: count down; 1: count up
7	Reserved	-
6:4	CKDIV	Count clock selection 000: PCLK0 001: PCLK0/2 010: PCLK0/4 011: PCLK0/8 100: 101: PCLK0/64 110: PCLK0/256 111: PCLK0/1024 PCLK0/16
3:1	MODE	Counting mode 000: Sawtooth wave A mode 100: Triangle wave A mode 101: Triangle wave B mode

Page 296

Please do not set other values

0 START Counter start 0: Counter close; 1: Counter start

Note: This bit will automatically change to 0 when the software stop condition or hardware stop condition is valid

Page 297

12.3.7 Interrupt Control Register (TIMx_ICONR)

Address offset: 0x054

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24	twentynine twenty two	twenty one	twelve	nine	18	17	16
Reserved							

15 14 13 12 11 10 9	8	7	6	5	4	3	2	1	0
Reserved	INTEN	INTEN	INTEN	Reserved	INTEN	INTEN	INTEN	INTEN	INTEN

Bit	mark	Function
31:9	Reserved	-
8	INTENDE	Dead time error interrupt enable 0: When the dead time is wrong, the interrupt is invalid 1: When the dead time is wrong, the interrupt is enabled
7	INTENUDF	Underflow interrupt enable 0: Underflow occurs during sawtooth wave or counts to valley point during triangle wave, this interruption is invalid 1: Underflow occurs during sawtooth wave or counts to valley point during triangle wave, this interrupt is enabled
6	INTENOVF	Overflow interrupt enable 0: Overflow occurs during sawtooth wave or counts to peak point during triangle wave, this interruption is invalid 1: When the overflow occurs in the sawtooth wave or the count reaches the peak point in the triangle wave, the interrupt is enabled
5:4	Reserved	-
3	INTEND	Count match interrupt enable D 0: When the GCMDR register is equal to the count value, the interrupt is invalid 1: When the GCMDR register is equal to the count value, the interrupt is enabled
2	INTENC	Count match interrupt enable C 0: When the GCMCR register is equal to the count value, the interrupt is invalid 1: When the GCMCR register is equal to the count value, the interrupt is enabled
1	INTENB	Count match interrupt enable B 0: When the GCMBR register is equal to the count value, or when a capture input event occurs, the interrupt is not enabled 1: When the GCMBR register is equal to the count value, or when a capture input event occurs, the interrupt enables can
0	INTENA	Count match interrupt enable A 0: When the GCMAR register is equal to the count value, or when a capture input event occurs, the interrupt is not available effect 1: When the GCMAR register is equal to the count value, or when a capture input event occurs, the interrupt enables can

Page 298**12.3.8 Port Control Register (TIMx_PCONR)**

Address offset: 0x058

Reset value: 0x0000 0000

31 30 29 28 27 26 25		twenty four twenty thirty two twenty one								19	18	17	16		
Reserved		DISVALB	DISSELB	OUTENB	PERCB	CMPCB	STASTPS	STPC	STAC	CAP					
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DISVALA	DISSELLA	OUTENA	PERCA	CMPCA	STASTPS	STPC	STAC	CAP					
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W					

Bit	mark	Function
31:29	Reserved	-
28:27	DISVALB	CHxB output status control 00: Forced output invalid condition When the selected condition in 0~3 is satisfied, the CHxB port will output normally 01: Forced output invalid condition 0~3 when the selected condition is satisfied, CHxB port outputs high impedance state 10: When the selected condition of forced output invalid condition 0~3 is satisfied, the CHxB port outputs low level 11: When the selected condition of forced output invalid condition 0~3 is satisfied, the CHxB port outputs high level
26:25	DISSELB	Forced output invalid condition selection B 00: select the forced output invalid condition 0; 01: select the forced output invalid condition 1 10: Select the forced output invalid condition 2; 11: Select the forced output invalid condition 3
twenty four	OUTENB	Output enable B 0: CHxB port output in Advanced Timer function is invalid 1: The CHxB port output in the Advanced Timer function is valid
23:22	PERCB	Port state setting B when the period value matches 00: When the counter count value is equal to the period value, the output of the CHxB port remains low 01: When the counter count value is equal to the period value, the CHxB port output is set to high level 10: When the counter count value is equal to the period value, the CHxB port output is set to the previous state 11: When the counter count value is equal to the period value, the CHxB port output is set to reverse level
21:20	CMPCB	Port status setting B when the comparison value matches 00: When the count value of the counter is equal to GCMBR, the output of the CHxB port remains low 01: When the counter count value is equal to GCMBR, the CHxB port output is set to high level 10: When the counter count value is equal to GCMBR, the CHxB port output is set to the previous state 11: When the counter count value is equal to GCMBR, the CHxB port output is set to reverse level
19	STASTPSB	Counting start stop port status selection B 0: When counting starts or stops, CHxB port output is determined by STACB and STPCB 1: When counting starts or stops, the CHxB port output is set to the previous state Note: The counting start here means the initial counting start or stop and then start; counting stop means the initial stop Stop or stop after counting starts

Page 299

18	STPCB	Counting stop port status setting B 0: When counting stops, CHxB port output is set to low level 1: When counting stops, CHxB port output is set to high level
17	STACB	Counting start port status setting B 0: When counting starts, CHxB port output is set to low level 1: When counting starts, CHxB port output is set to high level
16	CAPCB	Function mode selection B 0: Comparison output function; 1: Capture input function
15:13	Reserved	-
12:11	DISVALA	CHxA output status control 00: Forced output invalid condition When the selected condition in 0~3 is satisfied, the CHxA port will output normally 01: Forced output invalid conditions 0~3 when the selected conditions are met, CHxA port outputs high impedance state 10: When the selected condition of forced output invalid condition 0~3 is satisfied, the CHxA port outputs low level 11: When the selected condition of forced output invalid condition 0~3 is satisfied, the CHxA port outputs high level
10:9	DISSELA	Forced output invalid condition selection A 00: select the forced output invalid condition 0; 01: select the forced output invalid condition 1 10: Select the forced output invalid condition 2; 11: Select the forced output invalid condition 3
8	OUTENA	Output enable A 0: CHxA port output in Advanced Timer function is invalid 1: The CHxA port output in the Advanced Timer function is valid
7:6	PERCA	Port state setting A when the period value matches 00: When the counter count value is equal to the period value, the output of the CHxA port remains low 01: When the counter count value is equal to the period value, the CHxA port output is set to high level 10: When the counter count value is equal to the period value, the CHxA port output is set to the previous state 11: When the counter count value is equal to the period value, the CHxA port output is set to reverse level
5:4	CMPCA	Port status setting A when the comparison value matches 00: When the count value of the counter is equal to GCMAR, the output of the CHxA port remains low 01: When the counter count value is equal to GCMAR, the CHxA port output is set to high level 10: When the counter count value is equal to GCMAR, the CHxA port output is set to the previous state 11: When the counter value is equal to GCMAR, the CHxA port output is set to reverse level
3	STASTPSA	Count start stop port status selection A 0: When counting starts or stops, CHxA port output is determined by STACA and STPCA 1: When counting starts or stops, the CHxA port output is set to the previous state Note: The counting start here means the initial counting start or stop and then start; counting stop means the initial stop Stop or stop after counting starts
2	STPCA	Counting stop port status setting A 0: When counting stops, CHxA port output is set to low level; 1: When counting stops, CHxA port output is set to high level
1	STACA	Counting start port status setting A 0: When counting starts, CHxA port output is set to low level 1: When counting starts, CHxA port output is set to high level
0	CAPCA	Function mode selection A

HC32J110 Series User Manual Rev2.31

Page 300 of 527

Page 301

12.3.9 Cache Control Register (TIMx_BCONR)

Address offset: 0x05C

Reset value: 0x0000 0000

Bit	mark	Function
31:9	Reserved	-

8	BENP	Period value buffer transfer 0: Cache transfer is invalid 1: Buffer transmission enable (PERBR->PERAR)
7:3	Reserved	-
2	BENB	General comparison value buffer transfer B 0: Cache transfer is invalid 1: Cache transmission enable When comparing output function: (GCMDR->GCMBR); when capturing input function: (GCMBR->GCMDR)
1	Reserved	-
0	BENA	General comparison value buffer transfer A 0: Cache transfer is invalid 1: Cache transmission enable When comparing output function: (GCMCR->GCMAR); when capturing input function: (GCMAR->GCMCR)

Page 302**12.3.10 Dead zone control register (**TIMx_DCONR**)**

Address offset: 0x060

Reset value: 0x0000 0000

31 30 29 28 27 26 25	twentynine 28 27 26 25	twenty two 22	twenty one 20	19 18 17	16
		Reserved			
15 14 13 12 11 10	9	8	7	6	5 4 3 2 1 0
Reserved	SEPA	R/W	Reserved		DTCEN R/W

Bit	mark	Function
31:9	Reserved	-
8	SEPA	Separate setting 0: DTUAR and DTDAR are set separately 1: DTDAR value and DTUAR value are automatically equal
7:1	Reserved	-
0	DTCEN	Dead zone function 0: The dead zone function is invalid 1: The dead zone function is valid

Page 303**12.3.11 Filter control register (**TIMx_FCONR**)**

Address offset: 0x068

Reset value: 0x0000 0000

31	30	29	28	27 26 25	twenty four	22 21	20	19 18 17	16
Res.	NOFICKTD	NOFI	NOFICKTC	NOFI	NOFICKTB	NOFI	NOFICKTA	NOFI	
		ENTD Res.		ENTC Res.		ENTB Res.		ENTA	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11 10	9	8	7	6	5
					NOFICKGB	NOFI		NOFICKGA	NOFI
				Reserved			Res.		
					R/W	ENOB		R/W	ENOA

Bit	mark	Function		
31	Reserved	-		
30:29	NOFICKTD[1:0]	TRID port filter sampling reference clock selection		
		00: PCLK0	01: PCLK0/4	10: PCLK0/16
		PCLK0/64		11:
28	NOFIENTD	TRID port capture input filter enable, 0 is invalid; 1 is enabled		
27	Reserved	-		
26:25	NOFICKTC[1:0]	TRIC port filter sampling reference clock selection		
		00: PCLK0	01: PCLK0/4	10: PCLK0/16
		PCLK0/64		11:
twenty four	NOFIENTC	TRIC port capture input filter enable, 0 is invalid; 1 is enabled		
twenty three	Reserved	-		
22:21	NOFICKTB[1:0]	TRIB port filter sampling reference clock selection		
		00: PCLK0	01: PCLK0/4	10: PCLK0/16
		PCLK0/64		11:
20	NOFIENTB	TRIB port capture input filter enable, 0 is invalid; 1 is enabled		

19	Reserved	-			
18:17	NOFICKTA[1:0]	TRIA port filter sampling reference clock selection	00: PCLK0	01: PCLK0/4	10: PCLK0/16
		PCLK0/64			11:
16	NOFIENTA	TRIA port capture input filter enable, 0 is invalid; 1 is enabled			
15:7	Reserved	-			
6:5	NOFICKGB[1:0]	CHxIB port filter sampling reference clock selection	00: PCLK0	01: PCLK0/4	10: PCLK0/16
		PCLK0/64			11:
4	NOFIENGB	CHxIB port capture input filter enable, 0 is invalid; 1 is enabled			

Page 304

3	Reserved	-			
2:1	NOFICKGA[1:0]	CHxIA port filter sampling reference clock selection	00: PCLK0	01: PCLK0/4	10: PCLK0/16
		PCLK0/64			11:
0	NOFIENGA	CHxIA port capture input filter enable, 0 is invalid; 1 is enabled			

Notice:

- The TRIGA-D filter setting is only valid in TIM4, and invalid in Timer5/6.

Page 305**12.3.12 Effective Period Register (TIMx_VPERR)**

Address offset: 0x06C

Reset value: 0x0000 0000

31 30 29 28 27 26	25	twenty four	22 21	20	19	18	17	16
					PCNTS		PCNTE	
	Reserved				R/W		R/W	
15 14 13 12 11 10	9	8	7	6	5	4	3	2
						GE	GE	GE
	Reserved					PERI	PERIC	PERI
						R/W	R/W	R/W

Bit	mark	Function
31:21	Reserved	-
20:18	PCNTS	Effective period selection 000: The effective period selection function is invalid 001: valid once every 1 cycle 010: valid once every 2 cycles 011: valid once every 3 cycles 100: valid once every 4 cycles 101: Valid once every 5 cycles 110: Valid once every 6 cycles 111: Valid once every 7 cycles
17:16	PCNTE	Valid period counting condition selection 00: The effective period selection function is invalid 01: Sawtooth wave counting up, underflow point or triangle wave trough as counting condition 10: Sawtooth wave counting up, underflow point or triangle wave crest as counting condition 11: Sawtooth wave counting up and underflow points or triangular wave troughs and crests as counting conditions
15:4	Reserved	-
3	GEPERID	General signal valid period selection D 0: Valid period selection function is invalid; 1: Valid period selection function is enabled
2	GEPERIC	Common signal valid period selection C 0: Valid period selection function is invalid; 1: Valid period selection function is enabled
1	GEPERIB	General signal valid period selection B 0: Valid period selection function is invalid; 1: Valid period selection function is enabled
0	GEPERIA	General signal valid period selection A 0: Valid period selection function is invalid; 1: Valid period selection function is enabled

Page 306

12.3.13 Status Flag Register (TIMx_STFLR)

Address offset: 0x070

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24 23 22		twenty on 20	19	18	17
DIR							VPERNUM	R				
R									Reserved			
15	14	13	12	11	10	9	8	7	6	5	4	3 2 1 0
							DTEF UDFF OVFF			CMD	CMC	CMB CMA
							R/W	R/W	R/W	Reserved	R/W	R/W R/W R/W

Bit	mark	Function
31	DIRF	Counting direction 0: count down 1: Increasing counting
30:24	Reserved	-
23:21	VPERNUM	Number of cycles When the effective cycle selection function is enabled, the number of cycles after counting
20:9	Reserved	-
8	DTEF	Dead time error 0: No dead time error occurred; 1: Dead time error occurred
7	UDFF	Underflow matching 0: No sawtooth wave underflow occurs or triangle wave counts to the valley point 1: Sawtooth wave underflow occurs or triangle wave counts to the valley point
6	OVFF	Overflow matching 0: No sawtooth wave overflow occurs or triangle wave counts to the peak point 1: Sawtooth wave overflow occurs or triangle wave counts to the peak point
5:4	Reserved	-
3	CMDF	Count match D 0: The value of the GCMDR register is not equal to the count value; 1: The value of the GCMDR register and the count Equal value
2	CMCF	Count match C 0: GCMCR register value and count value are not equal; 1: GCMCR register value and count value equal
1	CMBF	Count match B 0: The value of the GCMBR register is not equal to the count value, and the CHxB capture completion action has not occurred 1: The value of the GCMBR register is equal to the count value, or the CHxB capture completion action occurs
0	CMAF	Count match A 0: The value of the GCMAR register is not equal to the count value, and the CHxA capture completion action has not occurred

Page 308**12.3.14 Hardware start event selection register (**TIMx_HSTAR**)**

Address offset: 0x074

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty	four	twenty	thirty	twenty	twenty	one	19	18	17	16
STARTS																	
R/W																	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	HSTA	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Bit	mark	Function
31	STARTS	<p>Hardware boot enable</p> <p>0: Invalid hardware startup 1: The hardware startup is valid</p> <p>Note: When the hardware startup is valid, the setting of <i>SSTAR</i> is invalid</p>
30:16	Reserved-	
15	HSTA15	<p>Hardware start condition 15: sampling on the TIMTRID port to the falling edge</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
14	HSTA14	<p>Hardware start condition 14: The rising edge is sampled on the TIMTRID port</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
13	HSTA13	<p>Hardware start condition 13: sampling on the TIMTRIC port to the falling edge</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
12	HSTA12	<p>Hardware start condition 12: The rising edge is sampled on the TIMTRIC port</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
11	HSTA11	<p>Hardware start condition 11: sampling on the TIMTRIB port to the falling edge</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
10	HSTA10	<p>Hardware start condition 10: the rising edge is sampled on the TIMTRIB port</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
9	HSTA9	<p>Hardware start condition 9: TIMTRIA port is sampled to the falling edge</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>

Page 309

8	HSTA8	<p>Hardware start condition 8: The rising edge is sampled on the TIMTRIA port</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
7	HSTA7	<p>Hardware start condition 7: CHxB port is sampled to the falling edge</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
6	HSTA6	<p>Hardware start condition 6: CHxB port is sampled to rising edge</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
5	HSTA5	<p>Hardware start condition 5: CHxA port is sampled to the falling edge</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
4	HSTA4	<p>Hardware start condition 4: CHxA port is sampled to rising edge</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
3	HSTA3	<p>Hardware start condition 3: Event trigger 3 from AOS is valid</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
2	HSTA2	<p>Hardware start condition 2: Event trigger 2 from AOS is valid</p> <p>0: When the conditions match, the hardware startup is invalid 1: When the conditions match, the hardware startup is valid</p>
1	HSTA1	<p>Hardware start condition 1: Event trigger 1 from AOS is valid</p> <p>0: When the conditions match, the hardware startup is invalid</p>

		1: When the conditions match, the hardware startup is valid
0	HSTA0	Hardware start condition 0: Event trigger from AOS 0 is valid
		0: When the conditions match, the hardware startup is invalid
		1: When the conditions match, the hardware startup is valid

Page 310**12.3.15 Hardware stop event selection register (**TIMx_HSTPR**)**

Address offset: 0x078

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty fourty thirty twenty twenty o@0	19	18	17	16
STOPs											
R/W											
15	14	13	12	11	10	9	8	7	6	5	4
HSTP	HSTA	HSTP	HSTP	HSTP	HSTP	HSTP	HSTP	HSTP	HSTP	HSTP	HSTP
15	14	13	12	11	10	9	8	7	6	5	4
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	mark	Function
31	STOPs	Hardware stop enable 0: hardware stop is invalid 1: Hardware stop is valid Note: When the hardware stop is valid, the software stop setting is invalid
30:16	Reserved-	
15	HSTP15	Hardware stop condition 15: sampling on the TIMTRID port to the falling edge 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
14	HSTP14	Hardware stop condition 14: The rising edge is sampled on the TIMTRID port 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
13	HSTP13	Hardware stop condition 13: sampling on the TIMTRIC port to the falling edge 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
12	HSTP12	Hardware stop condition 12: The rising edge is sampled on the TIMTRIC port

		0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
11	HSTP11	Hardware stop condition 11: sampling on the TIMTRIB port to the falling edge 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
10	HSTP10	Hardware stop condition 10: the rising edge is sampled on the TIMTRIB port 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
9	HSTP9	Hardware stop condition 9: sampling on the TIMTRIA port to the falling edge 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid

Page 311

8	HSTP8	Hardware stop condition 8: The rising edge is sampled on the TIMTRIA port 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
7	HSTP7	Hardware stop condition 7: CHxB port is sampled to the falling edge 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
6	HSTP6	Hardware stop condition 6: CHxB port is sampled to rising edge 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
5	HSTP5	Hardware stop condition 5: CHxA port is sampled to the falling edge 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
4	HSTP4	Hardware stop condition 4: CHxA port is sampled to rising edge 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
3	HSTP3	Hardware stop condition 3: Event trigger 3 from AOS is valid 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
2	HSTP2	Hardware stop condition 2: Event trigger 2 from AOS is valid 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
1	HSTP1	Hardware stop condition 1: Event trigger 1 from AOS is valid 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid
0	HSTP0	Hardware stop condition 0: Event trigger from AOS 0 is valid 0: When the conditions match, the hardware stop is invalid 1: When the conditions match, the hardware stops valid

Page 312

12.3.16 Hardware clear event selection register (**TIMx_HCELRL**)

Address offset: 0x07C

Reset value: 0x0000 0000

Bit	mark	Function
31	STARTS	<p>Hardware clear enable</p> <p>0: hardware clearing is invalid</p> <p>1: Hardware clearing is valid</p> <p>Note: When the hardware reset is valid, the software reset setting is invalid</p>
30:16	Reserved-	
15	HCEL15	<p>Hardware clear condition 15: TIMTRID port is sampled to the falling edge</p> <p>0: When the condition is matched, the hardware clearing is invalid</p> <p>1: When the conditions match, the hardware clearing is valid</p>
14	HCEL14	<p>Hardware clear condition 14: The rising edge is sampled on the TIMTRID port</p> <p>0: When the condition is matched, the hardware clearing is invalid</p> <p>1: When the conditions match, the hardware clearing is valid</p>
13	HCEL13	<p>Hardware clear condition 13: The sample on the TIMTRIC port reaches the falling edge</p> <p>0: When the condition is matched, the hardware clearing is invalid</p> <p>1: When the conditions match, the hardware clearing is valid</p>
12	HCEL12	<p>Hardware clear condition 12: The rising edge is sampled on the TIMTRIC port</p> <p>0: When the condition is matched, the hardware clearing is invalid</p> <p>1: When the conditions match, the hardware clearing is valid</p>
11	HCEL11	<p>Hardware clear condition 11: sampling on the TIMTRIB port to the falling edge</p> <p>0: When the condition is matched, the hardware clearing is invalid</p> <p>1: When the conditions match, the hardware clearing is valid</p>
10	HCEL10	<p>Hardware clearing condition 10: The rising edge is sampled on the TIMTRIB port</p> <p>0: When the condition is matched, the hardware clearing is invalid</p> <p>1: When the conditions match, the hardware clearing is valid</p>
9	HCEL9	<p>Hardware clear condition 9: sampling on the TIMTRIA port to the falling edge</p> <p>0: When the condition is matched, the hardware clearing is invalid</p> <p>1: When the conditions match, the hardware clearing is valid</p>

8	HCEL8	Hardware clear condition 8: The rising edge is sampled on the TIMTRIA port 0: When the condition is matched, the hardware clearing is invalid 1: When the conditions match, the hardware clearing is valid
7	HCEL7	Hardware clear condition 7: CHxB port is sampled to the falling edge 0: When the condition is matched, the hardware clearing is invalid 1: When the conditions match, the hardware clearing is valid
6	HCEL6	Hardware clear condition 6: CHxB port is sampled to rising edge 0: When the condition is matched, the hardware clearing is invalid 1: When the conditions match, the hardware clearing is valid
5	HCEL5	Hardware clear condition 5: CHxA port is sampled to the falling edge 0: When the condition is matched, the hardware clearing is invalid 1: When the conditions match, the hardware clearing is valid
4	HCEL4	Hardware clear condition 4: CHxA port is sampled to rising edge 0: When the condition is matched, the hardware clearing is invalid 1: When the conditions match, the hardware clearing is valid
3	HCEL3	Hardware clear condition 3: Event trigger 3 from AOS is valid 0: When the condition is matched, the hardware clearing is invalid 1: When the conditions match, the hardware clearing is valid
2	HCEL2	Hardware clearing condition 2: Event trigger 2 from AOS is valid 0: When the condition is matched, the hardware clearing is invalid 1: When the conditions match, the hardware clearing is valid
1	HCEL1	Hardware clear condition 1: Event trigger 1 from AOS is valid 0: When the condition is matched, the hardware clearing is invalid 1: When the conditions match, the hardware clearing is valid
0	HCEL0	Hardware clear condition 0: Event trigger from AOS 0 is valid 0: When the condition is matched, the hardware clearing is invalid 1: When the conditions match, the hardware clearing is valid

12.3.17 Hardware capture A event selection register (**TIMx_HCPAR)**

Address offset: 0x080

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty	twenty	twenty	twenty	o20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCPA	HCPA	HCPA	HCPA	HCPA	HCPA	HCPA	HCPA	HCPA	HCPA	HCPA	HCPA	HCPA	HCPA	HCPA	HCPA

R/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/W

Bit	mark	Function
-----	------	----------

31:16	Reserved-	
15	HCPA15	Hardware capture A condition 15: TIMTRID port is sampled to the falling edge 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
14	HCPA14	Hardware capture A condition 14: The rising edge is sampled on the TIMTRID port 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
13	HCPA13	Hardware capture A condition 13: sampling on the TIMTRIC port to the falling edge 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
12	HCPA12	Hardware capture A condition 12: The rising edge is sampled on the TIMTRIC port 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
11	HCPA11	Hardware capture A condition 11: TIMTRIB port is sampled to the falling edge 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
10	HCPA10	Hardware capture A condition 10: TIMTRIB port is sampled to the rising edge 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
9	HCPA9	Hardware capture A condition 9: TIMTRIA port is sampled to the falling edge 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
8	HCPA8	Hardware capture A condition 8: the rising edge is sampled on the TIMTRIA port 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
7	HCPA7	Hardware capture A condition 7: CHxB port is sampled to the falling edge 0: When conditions match, hardware capture A is invalid

Page 315

6	HCPA6	1: When the conditions match, the hardware capture A is valid Hardware capture A condition 6: CHxB port is sampled to rising edge 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
5	HCPA5	Hardware capture A condition 5: CHxA port is sampled to the falling edge 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
4	HCPA4	Hardware capture A condition 4: CHxA port is sampled to the rising edge 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
3	HCPA3	Hardware capture A condition 3: Event trigger 3 from AOS is valid 0: When conditions match, hardware capture A is invalid 1: When the conditions match, the hardware capture A is valid
2	HCPA2	Hardware capture A condition 2: Event trigger 2 from AOS is valid

		0: When conditions match, hardware capture A is invalid
		1: When the conditions match, the hardware capture A is valid
1	HCPA1	Hardware capture A condition 1: Event trigger 1 from AOS is valid
		0: When conditions match, hardware capture A is invalid
		1: When the conditions match, the hardware capture A is valid
0	HCPA0	Hardware capture A condition 0: Event trigger from AOS 0 is valid
		0: When conditions match, hardware capture A is invalid
		1: When the conditions match, the hardware capture A is valid

HC32L110 Series User Manual Rev2.31

Page 315 of 527

Page 316

12.3.18 Hardware capture B event selection register (TIMx_HCPBR)

Address offset: 0x084

Reset value: 0x0000 0000

Bit	mark	Function
31:16	Reserved-	
15	HCPB15	Hardware capture B condition 15: sampling on the TIMTRID port to the falling edge 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
14	HCPB14	Hardware capture B condition 14: the rising edge is sampled on the TIMTRID port 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid

13	HCPB13	Hardware capture B condition 13: sampling on the TIMTRIC port to the falling edge 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
12	HCPB12	Hardware capture B condition 12: the rising edge is sampled on the TIMTRIC port 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
11	HCPB11	Hardware capture B condition 11: sampling on the TIMTRIB port to the falling edge 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
10	HCPB10	Hardware capture B condition 10: the rising edge is sampled on the TIMTRIB port 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
9	HCPB9	Hardware capture B condition 9: sampling on the TIMTRIA port to the falling edge 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
8	HCPB8	Hardware capture B condition 8: The rising edge is sampled on the TIMTRIA port 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
7	HCPB7	Hardware capture B condition 7: CHxB port is sampled to the falling edge 0: When conditions match, hardware capture B is invalid

Page 317

6	HCPB6	1: When the conditions match, the hardware capture B is valid Hardware capture B condition 6: CHxB port is sampled to the rising edge 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
5	HCPB5	Hardware capture B condition 5: CHxA port is sampled to the falling edge 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
4	HCPB4	Hardware capture B condition 4: CHxA port is sampled to rising edge 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
3	HCPB3	Hardware capture B condition 3: Event trigger 3 from AOS is valid 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
2	HCPB2	Hardware capture B condition 2: Event trigger 2 from AOS is valid 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
1	HCPB1	Hardware capture B condition 1: Event trigger 1 from AOS is valid 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid
0	HCPB0	Hardware capture B condition 0: Event trigger from AOS 0 is valid 0: When conditions match, hardware capture B is invalid 1: When the conditions match, the hardware capture B is valid

Page 318

12.3.19 Hardware increment event selection register (TIMx_HCUPR)

Address offset: 0x088

Reset value: 0x0000 0000

Bit	mark	Function
31:20		Reserved-
19		HCUP19 Hardware increment condition: Event trigger 3 from AOS is valid 0: When the conditions match, the hardware increment is invalid 1: When the conditions match, the hardware increment is valid
18		HCUP18 hardware increment condition: event trigger 2 from AOS is valid 0: When the conditions match, the hardware increment is invalid 1: When the conditions match, the hardware increment is valid
17		HCUP17 Hardware increment condition: Event trigger 1 from AOS is valid 0: When the conditions match, the hardware increment is invalid 1: When the conditions match, the hardware increment is valid
16		HCUP16 hardware increment condition: event trigger from AOS 0 is valid 0: When the conditions match, the hardware increment is invalid 1: When the conditions match, the hardware increment is valid
15		HCUP15 hardware ramp-up condition: TIMTRID port is sampled to the falling edge 0: When the conditions match, the hardware increment is invalid 1: When the conditions match, the hardware increment is valid
14		HCUP14 hardware step-up condition: TIMTRID port is sampled to the rising edge 0: When the conditions match, the hardware increment is invalid 1: When the conditions match, the hardware increment is valid
13		HCUP13 hardware ramp-up condition: sampling on the TIMTRIC port to the falling edge 0: When the conditions match, the hardware increment is invalid 1: When the conditions match, the hardware increment is valid

- 12 HCUP12 hardware step-up condition: the rising edge is sampled on the TIMTRIC port
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid

Page 319

- 11 HCUP11 hardware progressive addition condition: TIMTRIB port is sampled to the falling edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid
- 10 HCUP10 hardware increment condition: sampling on the TIMTRIB port to rising edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid
- 9 HCUP9 Hardware ramp-up condition: TIMTRIA port is sampled to the falling edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid
- 8 HCUP8 Hardware step-up condition: TIMTRIA port is sampled to the rising edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid
- 7 HCUP7 Hardware increment condition: when the CHxB port is high, the CHxA port is sampled to the falling edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid
- 6 HCUP6 Hardware increment condition: When the CHxB port is high, the CHxA port is sampled to the rising edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid
- 5 HCUP5 Hardware increment condition: when the CHxB port is low, the CHxA port is sampled to the falling edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid
- 4 HCUP4 Hardware increment condition: when the CHxB port is low, the CHxA port is sampled to the rising edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid
- 3 HCUP3 Hardware increment condition: when the CHxA port is high, the CHxB port is sampled to the falling edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid
- 2 HCUP2 Hardware increment condition: When the CHxA port is high, the CHxB port is sampled to the rising edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid
- 1 HCUP1 Hardware increment condition: when the CHxA port is low, the CHxB port is sampled to the falling edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid
- 0 HCUP0 Hardware increment condition: When the CHxA port is low, the CHxB port is sampled to the rising edge
 0: When the conditions match, the hardware increment is invalid
 1: When the conditions match, the hardware increment is valid

12.3.20 Hardware decrement event selection register (**TIMx_HCDOR**)

Address offset: 0x08C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	26	25	20	19	18	17	16		
										HCD	HCD	HCD	HCD		
										o	o	o	o		
										R/W	R/W	R/W	R/W		
											17	16			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HCD															
o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	
R/W															

Bit	mark	Function
-----	------	----------

- 31:20 Reserved-
- 19 HCDO19 hardware decrement condition: event trigger 3 from AOS is valid
 - 0: When the conditions match, the hardware decrement is invalid
 - 1: When the conditions match, the hardware decrement is valid
- 18 HCDO18 hardware decrement condition: event trigger 2 from AOS is valid
 - 0: When the conditions match, the hardware decrement is invalid
 - 1: When the conditions match, the hardware decrement is valid
- 17 HCDO17 hardware decrement condition: event trigger 1 from AOS is valid
 - 0: When the conditions match, the hardware decrement is invalid
 - 1: When the conditions match, the hardware decrement is valid
- 16 HCDO16 hardware decrement condition: event trigger from AOS 0 is valid
 - 0: When the conditions match, the hardware decrement is invalid
 - 1: When the conditions match, the hardware decrement is valid
- 15 HCDO15 hardware decrement condition: TIMTRID port is sampled to the falling edge
 - 0: When the conditions match, the hardware decrement is invalid
 - 1: When the conditions match, the hardware decrement is valid
- 14 HCDO14 hardware decrement condition: TIMTRID port is sampled to rising edge
 - 0: When the conditions match, the hardware decrement is invalid
 - 1: When the conditions match, the hardware decrement is valid
- 13 HCDO13 hardware decrement condition: sampling on TIMTRIC port to falling edge
 - 0: When the conditions match, the hardware decrement is invalid
 - 1: When the conditions match, the hardware decrement is valid
- 12 HCDO12 hardware decrement condition: sampling on the TIMTRIC port to rising edge
 - 0: When the conditions match, the hardware decrement is invalid
 - 1: When the conditions match, the hardware decrement is valid

- 11 HCDO11 hardware decrement condition: TIMTRIB port is sampled to the falling edge
 - 0: When the conditions match, the hardware decrement is invalid
 - 1: When the conditions match, the hardware decrement is valid
- 10 HCDO10 hardware decrement condition: TIMTRIB port is sampled to rising edge
 - 0: When the conditions match, the hardware decrement is invalid

9	HCDO9	1: When the conditions match, the hardware decrement is valid Hardware decrement condition: TIMTRIA port is sampled to the falling edge 0: When the conditions match, the hardware decrement is invalid 1: When the conditions match, the hardware decrement is valid
8	HCDO8	Hardware decrement condition: TIMTRIA port is sampled to the rising edge 0: When the conditions match, the hardware decrement is invalid 1: When the conditions match, the hardware decrement is valid
7	HCDO7	Hardware decrement condition: when the CHxB port is high, the CHxA port is sampled to the falling edge 0: When the conditions match, the hardware decrement is invalid 1: When the conditions match, the hardware decrement is valid
6	HCDO6	Hardware decrement condition: when the CHxB port is high, the CHxA port is sampled to the rising edge 0: When the conditions match, the hardware decrement is invalid 1: When the conditions match, the hardware decrement is valid
5	HCDO5	Hardware decrement condition: when the CHxB port is low, the CHxA port is sampled to the falling edge 0: When the conditions match, the hardware decrement is invalid 1: When the conditions match, the hardware decrement is valid
4	HCDO4	Hardware decrement condition: when the CHxB port is low, the CHxA port is sampled to the rising edge 0: When the conditions match, the hardware decrement is invalid 1: When the conditions match, the hardware decrement is valid
3	HCDO3	Hardware decrement condition: when the CHxA port is high, the CHxB port is sampled to the falling edge 0: When the conditions match, the hardware decrement is invalid 1: When the conditions match, the hardware decrement is valid
2	HCDO2	Hardware decrement condition: when the CHxA port is high, the CHxB port is sampled to the rising edge 0: When the conditions match, the hardware decrement is invalid 1: When the conditions match, the hardware decrement is valid
1	HCDO1	Hardware decrement condition: when the CHxA port is low, the CHxB port is sampled to the falling edge 0: When the conditions match, the hardware decrement is invalid 1: When the conditions match, the hardware decrement is valid
0	HCDO0	Hardware decrement condition: when the CHxA port is low, the CHxB port is sampled to the rising edge 0: When the conditions match, the hardware decrement is invalid 1: When the conditions match, the hardware decrement is valid

Page 322

12.3.21 Software synchronization start register (**TIMx_SSTAR**)

Address offset: 0x3F4

Reset value: 0x0000 0000

Bit	mark	Function
31:3	Reserved	
2	SSTA2	Timer6 software start 0: Software startup is invalid 1: Software startup enable
1	SSTA1	Timer5 software start 0: Software startup is invalid 1: Software startup enable
0	SSTA0	Timer4 software start 0: Software startup is invalid 1: Software startup enable

Page 323**12.3.22 Software synchronization stop register (**TIMx_SSTPR**)**

Address offset: 0x3F8

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twent	twent	twent	twent	o@0	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
R/WR/WR/W															

Bit	mark	Function
31:3	Reserved	
2	SSTP2	Timer6 software stop 0: Software stop is invalid 1: Software stop enable
1	SSTP1	Timer5 software stop 0: Software stop is invalid 1: Software stop enable

0	SSTP0	Timer4 software stop 0: Software stop is invalid 1: Software stop enable
---	-------	--

Page 324**12.3.23 Software Synchronous Clear Register (**TIMx_SCLRR**)**

Address offset: 0x3FC

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twent	four	thirt	twent	twent	o20	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
SCLR2 SCLR1 SCLR0																
R/WR/WR/W																

Bit	mark	Function
31:3	Reserved	
2	SCLR2	Timer6 software cleared 0: Software clearing is invalid 1: Software clearing enable
1	SCLR1	Timer5 software cleared 0: Software clearing is invalid 1: Software clearing enable
0	SCLR0	Timer4 software cleared 0: Software clearing is invalid 1: Software clearing enable

Page 325**12.3.24 Interrupt Flag Register (TIMx_IFR)**

Address offset: 0x100

Reset value: 0x0000 0000

31	30	29 28	27	26	25	twenty four	twenty three	twenty two	twenty one	0	19	18	17	16
Reserved														
15	14 13 12		11	10	9	8	7	6	5	4 3	2	1	0	
SAMHF	SAMLF					DTEF	UDFF	OVFF		CMDF	CMCF	CMBF	CMAF	
RO	RO	Reserved				RO	RO	RO	Reserved	RO	RO	RO	RO	
Bit	mark	Name	Function											
31:16	Reserved	-												
15	SAMHF	CHxA/B port high state interrupt flag												
		0: There is no high level on the CHxA and CHxB ports at the same time												
		1: High level appears on both CHxA and CHxB ports at the same time												
14	SAMLF	CHxA/B port low state interrupt flag												
		0: There is no low level on the CHxA and CHxB ports at the same time												
		1: Low level appears on both CHxA and CHxB ports at the same time												
13:9	Reserved	-												
8	DTEF	Dead time error interrupt flag												
		0: No dead time error occurred; 1: Dead time error occurred												
7	UDFF	Underflow match interrupt flag												
		0: No sawtooth wave underflow occurs or triangle wave counts to the valley point												
		1: Sawtooth wave underflow occurs or triangle wave counts to the valley point												
6	OVFF	Overflow match interrupt flag												
		0: No sawtooth wave overflow occurs or triangle wave counts to the peak point												
		1: Sawtooth wave overflow occurs or triangle wave counts to the peak point												
5:4	Reserved	-												
3	CMDF	Count match D interrupt flag												
		0: The value of the GCMDR register is not equal to the count value; 1: The value of the GCMDR register and the count Equal value												
2	CMCF	Count match C interrupt flag												
		0: The value of the GCMCR register is not equal to the count value; 1: The value of the GCMCR register and the count Equal value												
1	CMBF	Count match B interrupt flag												

0: The value of the GCMBR register is not equal to the count value, and the CHxB capture completion action has not occurred

1: The value of the GCMBR register is equal to the count value, or the CHxB capture completion action occurs

0

CMAF

Count match A interrupt flag

0: The value of the GCMAR register is not equal to the count value, and the CHxA capture completion action has not occurred

Page 326

1: The value of the GCMAR register is equal to the count value, or the CHxA capture completion action occurs

12.3.25 Interrupt Flag Clear Register (TIMx_ICLR)

Address offset: 0x104

Reset value: 0x0000 0000

Bit	mark	Function
31:16	Reserved	-
15	SAMHC	CHxA/B port high state interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
14	SAMLC	CHxA/B port low state interrupt flag is cleared, writing 1 is invalid, writing 0 to clear the corresponding interrupt
13:9	Reserved	-
8	DTEC	Dead time error interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
7	UDFC	Underflow match interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
6	OVFC	Overflow match interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
5:4	Reserved	-
3	CMDC	Count match D interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
2	CMCC	Count match C interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
1	CMBC	Count match B interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt
0	CMAC	Count match A interrupt flag is cleared, writing 1 is invalid, writing 0 clears the corresponding interrupt

12.3.26 Spread spectrum and interrupt trigger selection (TIMx CR)

Address offset: 0x108

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16	
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
DITENS DITENB DITNA UDFE OVFE							CMDE CMCE CMBE CMAE								
R/WR/WR/WR/WR/W							Reserved								
R/WR/WR/WR/W															

Bit	mark	Function
31:11	Reserved	-
10	DITENS	PWM spread frequency counting selection 0: select underflow, 1: select overflow
9	DITENB	PWM channel B spread spectrum enable 0: Enable is invalid, 1: Enable is valid, change the output delay of PWM every cycle
9	DITENA	PWM channel A spread spectrum enable 0: Enable is invalid, 1: Enable is valid, change the output delay of PWM every cycle
7	UDFE	Underflow match enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS_i_tirg
6	OVFE	Overflow match enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS_i_tirg
5:4	Reserved	-
3	CMDE	Count match D enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS_i_tirg
2	CMCE	Count match C enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS_i_tirg
1	CMBE	Count match B enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS_i_tirg
0	CMAE	Count match A enable trigger ADC 0: Enable is invalid, 1: Enable is valid, this interrupt can control ADC/AOS_i_tirg

12.3.27 AOS selection control register (TIMx_AOSSR)

Address offset: 0x110

Reset value: 0x0000 0000

Timer4/5/6 uses the same physical register. After any timer is changed, the

The value will change at the same time.

31	30	29	28	27	26	25	twenty four	twenty three	21	20	19	18	17	16	
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
R/WR/WR/WR/W															

User manual

Reserved SMH2 SMH1 SMH0 SML2 SML1 SML0 SOFTB Reserved BFILTEN BFILTS FSAME FBRAKE

Bit	mark	Function
31:14	Reserved	-
13	SMH2	Channel 2 same height selection 0: The selection is invalid, 1: The selection is valid, when the same height appears AOS_i_odis[1]
12	SMH1	Channel 1 same height selection 0: The selection is invalid, 1: The selection is valid, when the same height appears AOS_i_odis[1]
11	SMH0	Channel 0 same height selection 0: The selection is invalid, 1: The selection is valid, when the same height appears AOS_i_odis[1]
10	SML2	Channel 2 with low selection 0: selection is invalid, 1: selection is valid, when the same low occurs AOS_i_odis[1]
9	SML1	Channel 1 with low selection 0: selection is invalid, 1: selection is valid, when the same low occurs AOS_i_odis[1]
8	SML0	Channel 0 with low selection 0: selection is invalid, 1: selection is valid, when the same low occurs AOS_i_odis[1]
7	SOFTBK	Software brake: write 1 to realize software brake
13	Reserved	-
4	BFILTEN	Port brake filter enable
3:2	BFILTS	Port brake filter clock selection
1	FSAME	Same high and same low brake mark, read only
0	FBRAKE	Port brake flag, read only

Page 330

12.3.28 AOS selection control register flag clear (**TIMx_AOSCL**)

Address offset: 0x114

Reset value: 0x0000 0000

Timer4/5/6 uses the same physical register. After any timer is changed, the

The value will change at the same time.

Bit	mark	Function
31:2	Reserved	-
1	FSAME	Same high and same low brake flag is cleared, write 0 to clear, write 1 is invalid, read is always 1
0	F BRAKE	Port brake flag is cleared, write 0 to clear, write 1 is invalid, read is always 1

Page 331**12.3.29 Port brake control register (TIMx_PTBKs)**

Address offset: 0x118

Reset value: 0x0000 0000

Timer4/5/6 uses the same physical register. After any timer is changed, the

The value will change at the same time.

31	30	29	28	27	26	25	twenty four	twenty thirty	twenty twenty	twenty one	19	18	17	16
Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1

R/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/WR/W

Bit	mark	Function
31:16	Reserved-	
15	EN15	P36 Brake port enable: 1 option, 0 invalid
14	EN14	P35 Brake port enable: 1 option, 0 invalid
13	EN13	P34 Brake port enable: 1 option, 0 invalid
12	EN12	P33 Brake port enable: 1 option, 0 invalid
11	EN11	P32 Brake port enable: 1 option, 0 invalid
10	EN10	P31 Brake port enable: 1 option, 0 invalid
9	EN9	P27 Brake port enable: 1 option, 0 invalid
8	EN8	P26 Brake port enable: 1 option, 0 invalid
7	EN7	P26 Brake port enable: 1 option, 0 invalid
6	EN6	P24 Brake port enable: 1 option, 0 invalid
5	EN5	P23 Brake port enable: 1 option, 0 invalid

4	EN4	P15 Brake port enable: 1 option, 0 invalid
3	EN3	P14 Brake port enable: 1 option, 0 invalid
2	EN2	P03 Brake port enable: 1 is selected, 0 is invalid
1	EN1	P02 Brake port enable: 1 option, 0 invalid
0	EN0	P01 Brake port enable: 1 is selected, 0 is invalid

Page 332**12.3.30 Port trigger control register (TIMx_TTRIG)**

Address offset: 0x11C

Reset value: 0x0000 0000

Timer4/5/6 uses the same physical register. After any timer is changed, the

The value will change at the same time.

31	30	29	28	27	26	25	twenty	fourty	thwenty	twenty	on	0	19	18	17	16
----	----	----	----	----	----	----	--------	--------	---------	--------	----	---	----	----	----	----

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIGDS				TRIGCS					TRIGBS			TRIGAS			
R/W				R/W					R/W			R/W			

Bit	mark	Function	-
31:16	Reserved		-
15:12	TRIGDS	TIMx trigger D port selection	
11:8	TRIGCS	TIMx trigger C port selection	
7:4	TRIGBS	TIMx trigger B port selection	
3:0	TRIGAS	TIMx trigger A port selection	

The control signal and port selection are as follows

0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111

P01	P02	P03	P15	P14	P23	P24	P25	P26	P27	P31	P32	P33	P34	P35	P36
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

Page 333

12.3.31 AOS trigger control register (TIMx_ITRIG)

Address offset: 0x120

Reset value: 0x0000 0000

Timer4/5/6 uses the same physical register. After any timer is changed, the

The value will change at the same time.

31	30	29	28	27	26	25	twenty four	thirty two	thirty two	thirty two	19	18	17	16	
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	IAOS3S				IAOS2S				IAOS1S			IAOS0S			
	R/W				R/W				R/W			R/W			

Bit	mark	Function
31:16	Reserved	-
15:12	IAOS3S	TIMx AOS3 trigger source selection
11:8	IAOS2S	TIMx AOS2 trigger source selection
7:4	IAOS1S	TIMx AOS1 trigger source selection
3:0	IAOS0S	TIMx AOS0 trigger source selection

The control signal (IAOSxS) and interrupt source are selected as follows (x=0,1,2,3)

0000	0001	0010	0011	0100	0101	0110	0111
TIM0_INT	TIM1_INT	TIM2_INT	LPTIMER_INT	TIM4_INTS	TIM5_INTS	TIM6_INTS	UART0_INT
1000	1001	1010	1011	1100	1101	1110	1111
UART1_INT	LPUART_INT	VCI_INT	VC2_INT	RTC_INT	PCA_INT	SPI_INT	ADC_INT

12.3.32 Port brake polarity control register (TIMx_PTBKPx)

Address offset: 0x124

Reset value: 0x0000 0000

Timer4/5/6 uses the same physical register. After any timer is changed, the

The value will change at the same time.

Bit	mark	Function
31:16	Reserved-	
15	POL15	P36 Brake port polarity selection: 1 low level is valid, 0 high level is valid
14	POL14	P35 Brake port polarity selection: 1 low level is valid, 0 high level is valid
13	POL13	P34 Brake port polarity selection: 1 low level is valid, 0 high level is valid
12	POL12	P33 Brake port polarity selection: 1 low level is valid, 0 high level is valid
11	POL11	P32 Brake port polarity selection: 1 low level is valid, 0 high level is valid
10	POL10	P31 Brake port polarity selection: 1 low level is valid, 0 high level is valid
9	POL9	P27 Brake port polarity selection: 1 low level is valid, 0 high level is valid
8	POL8	P26 Brake port polarity selection: 1 low level is valid, 0 high level is valid
7	POL7	P26 Brake port polarity selection: 1 low level is valid, 0 high level is valid
6	POL6	P24 Brake port polarity selection: 1 low level is valid, 0 high level is valid
5	POL5	P23 Brake port polarity selection: 1 low level is valid, 0 high level is valid
4	POL4	P15 Brake port polarity selection: 1 low level is valid, 0 high level is valid
3	POL3	P14 Brake port polarity selection: 1 low level is valid, 0 high level is valid
2	POL2	P03 Brake port polarity selection: 1 low level is valid, 0 high level is valid
1	POL1	P02 Brake port polarity selection: 1 low level is valid, 0 high level is valid
0	POL0	P01 Brake port polarity selection: 1 low level is valid, 0 high level is valid

13 real time clock (RTC)

13.1 Introduction to Real Time Clock

The real-time clock/calendar provides information of seconds, minutes, hours, days, weeks, months, and years. The number of days in a month and the auto-adjust. Clock operation can use AM/PM register bit to decide to use 24 or 12 hour format. Table 14-1 shows its basic characteristics.

	Off-chip low-speed crystal XTL (32.768kHz)
Clock source	On-chip low-speed oscillator RCL (32kHz, 1% accuracy)
	Off-chip high-speed crystal XTH
	Calculate seconds, minutes, hours, days, weeks, months, and years between 00 and 99 years
	Automatic leap year adjustment
basic skills	Configurable to 24 or 12 hour format
	Programmable start or stop
	With alarm function
	With high-precision 1Hz square wave signal output
Interrupt	With periodic interrupt
	With alarm interrupt

Table 13-1 Basic features of RTC

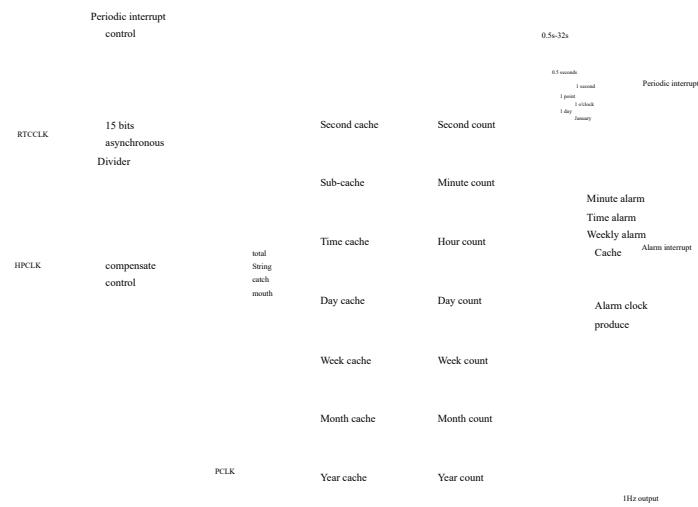


Figure 13-1 RTC block diagram

13.2 Real-time clock function description

The clock source of the real-time clock can be configured as an external low-speed crystal oscillator, an external high-speed crystal oscillator, and an internal low-speed crystal oscillator. The control registers CR0, CR1 and COMPEN are only controlled by power-on reset, other reset sources These three control registers cannot be reset. The power-on status of other data registers is uncertain and needs to be initialized after power-on. Affected by any reset.

All the date and time values written and read by the software are BCD codes, and there is no need to convert hexadecimal to decimal.

Any invalid date and time cannot be written, such as 32 days, 25 o'clock, 70 seconds, B month, etc.

13.2.1 Power-on settings

The RTC is reset once after power-on. When the system is not powered down, various external reset requests cannot be reset. RTC, RTC will always be in counting state. After power-on, set the initial value of the calendar, alarm setting, error compensation. After compensation, interruption, etc., start RTC.

13.2.2 RTC counting start setting

1. Set CR0.START=0, counting stops;
2. Set CR0.AMPM and CR0.PRDS, CR0.PRDY set time system and interrupt cycle;
3. Set CR1.CKSEL to select the timing clock of RTC;
4. Set the calendar counting registers for seconds, minutes, hours, weeks, days, months, and years;
5. When clock error compensation is needed, set the counting clock error compensation register COMPEN;
6. Clear the interrupt flags CR1.ALMF and CR1.PRDF, and enable the interrupt;
7. Set CR0.START=1 to start counting.

13.2.3 System Low Power Mode Switching

After the RTC count starts, if the system immediately switches to low power consumption mode, please perform one of the following confirmations. Then switch the mode.

The control register is in the system control register SYSCTRL1.RTC_LPW

1. After CR0.START=1 is set, the mode is switched after more than 2 RTC count clocks have elapsed.
2. After setting CR0.START=1, set CR1.WAIT=1, and query CR1.WAITF=1. Reset

CR1.WAIT=0, query CR1.WAITF=0 before switching the mode.

In RTC low power consumption mode, RTC registers cannot be read or written. In low power mode, RTC consumes less power flow.

There is no need to wait to switch low-power mode while RTC is running.

13.2.4 Read count register

There are three ways to read the count register:

Method 1: Reading method 1 at any time

1. Set CR1.WAIT=1 to stop the calendar register counting and enter the read-write mode;
2. Query until CR1.WAITF=1;
3. Read the second, minute, hour, week, day, month, and year count register values;
4. Set CR1.WAIT=0, the counter counts;
5. Query until CR1.WAITF=0.

Method 2: Reading method 2 at any time

1. Read the minute, hour, week, day, month, and year count register values;
2. Read the second counter register value;
3. Read the second counter register value again;
4. Determine whether the two second reading values are the same, start again from the first step with the difference, and end the same reading.

Mode 3: Interrupt reading mode

Read the second, minute, hour, week, day, month, and year count register values in the RTC periodic interrupt service. Because of interruption
The time from the occurrence to the next data change at least 0.5s.

13.2.5 Write to count register

1. Set CR1.WAIT=1 to stop the calendar register counting and enter the read-write mode;
2. Query until CR1.WAITF=1;
3. Write the second, minute, hour, week, day, month, and year count register values;
4. Set CR1.WAIT=0, the counter restarts counting. Note that all write operations must be completed within 1 second;
5. Query until CR1.WAITF=0.

When writing seconds, minutes, hours, weeks, days, months, and years in the RTC not-started mode, there is no need to wait for WAIT.

Note:

-Changing the second register in the counting mode will reset the second count, write minute, hour, week, day, month, and year count registers
The value does not affect the RTC count.

13.2.6 Alarm setting

1. Set CR1.ALMEN=0, the alarm is disabled;
2. Set CR1.ALMIIE=1, alarm interrupt permission;
3. Set the minute alarm clock ALMMIN, hour alarm clock ALMHOUR, and weekly alarm clock ALMEEK;
4. Set CR1.ALMEN=1, the alarm clock is allowed;
5. Wait for an interrupt to occur;
6. Since the alarm clock interrupt and the fixed cycle interrupt share the interrupt request signal, when CR1.ALMF=1, enter the alarm clock
Interrupt processing; otherwise, enter the fixed cycle interrupt processing.

13.2.7 1Hz output

RTC can choose to output three types of 1Hz clocks: general precision, higher precision and high precision. When the clock error compensation function is valid, output a high-precision 1Hz clock; when using PCLK with different frequencies, output a high-precision 1Hz clock
Bell. The system control register needs to be configured according to the PCLK frequency, among which,

The general precision 1Hz output setting is as follows: (no clock compensation)

1. Set CR0.START=0, counting stops;
2. RTC output pin setting;

3. CR0.1HZOE=1, clock output permission;
4. Set CR0.START=1 to start counting;
5. Wait for more than 2 counting cycles;
6. The 1Hz output starts.

The higher precision 1Hz output setting is as follows: (low speed compensation)

1. Set CR0.START=0, counting stops;
2. RTC output pin setting;
3. CR0.1HZOE=1, clock output permission;
4. Clock error compensation register COMPEN.CR compensation number setting;

5. Clock error compensation register COMPEN.EN=1, error compensation is valid;
6. Set CR0.START=1 to start counting;
7. Wait for more than 2 counting cycles;
8. The 1Hz output starts.

When high-precision 1Hz output is required, it is necessary to provide RTC on the basis of higher-precision output

4M, 6M, 8M, 12M, 16M, 20M, 24M, 32MHz high-speed PCLK clock, the output settings are as follows:

1. Set CR0.START=0, counting stops;
2. RTC output pin setting;
3. CR0.1HZOE=1, clock output permission;
4. CR0.1HZSEL=1, select to output high-precision 1Hz clock;
5. Configure high-speed clock compensation clock SYSCTRL1.RTC_FREQ_ADJUST
6. Compensation number setting of clock error compensation register COMPEN.CR[8:0];
7. Clock error compensation register COMPEN.EN=1, accuracy compensation is valid;
8. Set CR0.START=1 to start counting;
9. Wait for more than 2 counting cycles;
10. The 1Hz output starts.

13.2.8 Clock Error Compensation

Due to the error of the external crystal oscillator, when a high-precision counting result is required, the error needs to be compensated.

There are two compensation methods: the first is based on the error compensation of its own clock; the second is based on the error of the high-speed clock. Difference compensation.

Principle and calculation of error compensation based on its own clock:

Since the counter uses a 32.768KHz clock to count, if you need to compensate for the accuracy per second, you can only follow 32.768KHz integer period compensation, the smallest unit of compensation per second is $(1/32768)*10^{-6} = 30.5\text{ppm}$, no. The method meets the requirements of high precision.

Then, to achieve high-precision clock compensation under the count clock of 32.768KHz, it is necessary to adjust the algorithm. The maximum compensation period is expanded by 32 times. If the smallest unit that can only be compensated is 30.5ppm, the average

The compensation unit averaged to every second becomes $30.5\text{ppm}/32=0.96\text{ppm}$. Meet the high precision clock compensation requirements. and

Page 340

And compensation occurs in a relatively uniform range every 32 seconds. Therefore, a 5-digit decimal setting is introduced in this register.

Certainly.

example 1:

When the 1Hz clock is directly output in the default state, the compensation target value is calculated by measuring the accuracy of the clock.

Assuming the actual measured value is 0.9999888Hz, then:

Actual vibration frequency = $32768 \times 0.9999888 \approx 32767.63$

Compensation target = $(\text{Actual vibration frequency} - \text{target frequency}) / \text{target frequency} \times 10^6$

$$= (32767.96 - 32768) / 32768 \times 10^6$$

$$= -11.29\text{ppm}$$

according to

$$\text{CR[8:0]}: = \left(\begin{array}{c} \text{Compensation target value} \\ 10^6 \end{array} \right) [\text{ppm}] + 0001.00000 \text{ B}$$

Take 2's complement

If the compensation target value is -11.29ppm, calculate the corresponding register value as follows:

$$\text{CR[8:0]} = (-11.29 \times 2^{15} / 10^6) \text{ take 2's complement} + 0001.00000 \text{ B}$$

$$= (-0.37) \quad \text{Take 2's complement} + 0001.00000 \text{ B}$$

$$= 1111.10101 \text{ B} + 0001.00000 \text{ B}$$

$$= 0000.10101 \text{ B}$$

Principle and calculation of error compensation based on high-speed **24MHz** clock:

The calculation method of this method is the same as the error compensation based on the own clock. Due to the introduction of 4M-32MHz high-spee

Time, the error of 1/32768 seconds that would have been accumulated within 32 seconds at most can be dispersed to every 1 second.

Line minimum 0.96ppm (23 24MHz clock cycles) compensation to achieve an average high-precision 1Hz clock per second

Output.

13.3 RTC interrupt

RTC supports two types of interrupts. Alarm interruption, fixed cycle interruption. Alarm clock interrupt and fixed cycle interrupt share one Signal off.

13.3.1 RTC alarm interrupt

When CR1.ALMI=1, if the current calendar time and minute alarm register (ALMMIN), hour alarm register When (ALMHOUR) and weekly alarm register (ALM WEEK) are equal, the alarm interrupt is triggered.

13.3.2 RTC cycle interrupt

When ALMIE=1 in control register 1 (CR1), after the selected period occurs, a fixed-period wake-up interrupt is triggered. The alarm clock and the fixed cycle share interrupts, which are distinguished by the flag register bit.

13.4 RTC register description

Base address 0X40001400

https://translate.googleusercontent.com/translate_f

293/453

register	Offset address	describe
RTC_CR0	0X000	Control register 0
RTC_CR1	0X004	Control register 1
RTC_SEC	0X008	Second count register
RTC_MIN	0X00C	Sub-count register
RTC_HOUR	0X010	Hour counter register
RTC_WEEK	0X014	Week count register
RTC_DAY	0X018	Day count register
RTC_MON	0X01C	Month count register
RTC_YEAR	0X020	Year count register
RTC_ALMMIN	0X024	Sub-alarm register
RTC_ALMHOUR	0X028	Time alarm register
RTC_ALMWEEK	0X02C	Weekly alarm register
RTC_COMPEN	0X030	Clock error compensation register

Table 13-2 List of RTC Registers

13.4.1 Control Register 0 (RTC_CR0)

*Only power-on reset is valid for this register

Address offset: 0x000

Reset value 0x0000 0000

31	30	29	28	27	26	25	twenty	forty	thirty	twenty	twenty	0@0	19	18	17	16
Reserved																
15	14		13:8		7	6	5	4	3		2:0					
		PRDSEL		PRDX		START	HZ1SEL	HZ1OE		AMPM		PRDS				

Res.	R/W	R/W	R/W	R/W	Res.	R/W	R/W
Bit	symbol	Function description					
31:15	Reserved-						
14	PRDSEL 0: Use the periodic interrupt time interval set by PRDX						
	1: Use the periodic interrupt time interval set by PRDX						
13:8	PRDX	Set the time interval for generating periodic interrupts. The range that can be set is 0.5 seconds to 32 seconds, and the step is 0.5 seconds.					
	000000: 0.5 seconds						
	000001: 1 second						
	...						
	111110: 31.5 seconds						
	111111: 32 seconds						
7	START	0: Stop RTC counter					
	1: Enable RTC counter						
6	1HZSEL 0: Normal precision 1Hz output						
	1: High precision 1Hz output						
5	1HZOE	0: Disable 1Hz output					
	1: Enable 1Hz output						
4	Reserved-						
3	AMPM	0: 12-hour clock					
	1: 24-hour system						
2:0	PRDS	Set the time interval for generating interrupts:					
	000: Do not generate periodic interrupts						
	001: 0.5 seconds						
	010: 1 second						
	011: 1 minute						
	100: 1 hour						
	101: 1 day						
	11x: January						

Note: If you need to write the time interval for changing the cycle interruption when START=1, the operation steps are as follows:

step1, turn off RTC interrupt in NVIC;
 step2, change the time interval of periodic interruption;
 step3, clear the RTC interrupt flag;
 Step4, enable RTC interrupt.

Page 345**13.4.2 Control Register 1 (RTC_CR1)**

*Only power-on reset is valid for this register

Address offset: 0x004

Reset value 0X00000000

31	30	29	28	27	26 25	twenty fourty twenty twenty twenty off0	19	18	17	16
Reserved										

15:11	10:8	7	6	5	4	3	2	1	0
14	CKSEL	ALMEN	ALMIE	Res.	ALMF	PRDF	Res.	WAITF	WAIT
Reserved	R/W	R/W	R/W	RO	RO	RO	Res.	R/W	R/W

Bit symbol Function description

31:11 Reserved -

10:8 CKSEL RTC clock selection

00x: XTL 32.768k

01x: RCL 32k

100: XTH/128 (select this when the crystal oscillator is 4M)

101: XTH/256 (Select this when the crystal oscillator is 8M)

110: XTH/512 (select this when the crystal oscillator is 16M)

111: XTH/1024 (Select this when the crystal oscillator is 32M)

7 ALMEN 0: Disable alarm clock

1: Enable the alarm clock

Note: It is enabled when START=1 during calendar counting and ALMIE=1 interrupt permission

For ALMEN, turn off the system interrupt to prevent malfunction. Please clear the ALMF flag after enabling remove.

6	ALMIE	0: Disable alarm interrupt 1: Enable alarm interrupt
5	Reserved	-
4	ALMF	0: No alarm interruption occurred 1: Alarm interruption has occurred Note: This bit is only valid when ALMEN=1. When the alarm is matched, 32.768KHz is set to 1 after a clock. The flag is cleared when writing 0, and writing 1 is invalid.
3	PRDF	0: Periodic interruption has not occurred 1: Periodic interrupt has occurred Note: After a period interrupt occurs, this bit is set to 1. This flag is cleared when writing 0, and writing 1 is invalid.
2	Reserved	-
1	WAITF	0: Non-write/read state 1: Write/read status Note: The WAIT bit is a valid flag. Please confirm whether this bit is "1" before writing/reading. count

Page 346

During the counting process, wait until the WAIT bit is cleared to "0" before the bit is cleared to "0" after the writing is completed.

0	WAIT	0: Normal counting mode 1: Write/read mode Note: Please set this bit to "1" when writing/reading, because the counter is counting continuously, please wait for 1 second The write/read operation is completed within and the bit is cleared to "0".
---	------	---

Page 347**13.4.3 Second Count Register (RTC_SEC)**

Address offset: 0x008

Reset value: indefinite

31	30	29	28	27	26	25	twenty	twenty	thirty	twenty	twenty	on off	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
-																
Reserved										SECH	SECL				R/W	

Bit	symbol	Function description
31:7	Reserved	-
6:4	SECH	Tens of seconds count
3:0	SECL	Seconds count single place value

Represents 0-59 seconds and uses decimal counting. Please write the decimal 0-59 BCD code, when writing an error value, write The entered value will be ignored.

13.4.4 Sub-counting register (RTC_MIN)

Address offset: 0x00C

Reset value: indefinite

31	30	29	28	27	26	25	twenty	twenty	thirty	twenty	twenty	on off	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
-																
Reserved										MINH	MINL				R/W	

Bit	symbol	Function description
31:7	Reserved	-
6:4	MINH	Minute tens value
3:0	MINL	Minute count single place value

Represents 0-59 minutes and uses decimal counting. Please write the decimal 0-59 BCD code, when writing an error value, write The entered value will be ignored.

13.4.5 Hour counter register (RTC_HOUR)

Address offset: 0x010

Reset value: indefinite

Bit	symbol	Function description
31:6	Reserved	-
5:4	HOURH	Hour count tens value
3:0	HOURL	Hour count single place value

In 24-hour time format, it means 0-23 hours. In 12-hour time format, b5=0 means AM, then 01:12 means up
Noon; b5=1 means PM, then 21:32 means afternoon.

Please set the correct decimal 0:23 or 01:12,21:32 BCD code according to the value of AM/PM.

Values written out of the range will be ignored.

Refer to the following table for specific time indication:

24-hour clock

AMPM=1

12-hour clock

AMPM=0

time	Register representation	time	Register representation
00 o'clock	00H	AM 12 o'clock	12H
01 o'clock	01H	AM 01 o'clock	01H
02 o'clock	02H	AM 02 o'clock	02H
03 o'clock	03H	AM 03 o'clock	03H
04 o'clock	04H	AM 04 o'clock	04H
05 o'clock	05H	AM 05 o'clock	05H
06 o'clock	06H	AM 06 o'clock	06H
07 o'clock	07H	AM 07 o'clock	07H
08 o'clock	08H	AM 08 o'clock	08H
09 o'clock	09H	AM 09 o'clock	09H
10 o'clock	10H	AM 10 o'clock	10H
11 o'clock	11H	AM 11 o'clock	11H
12 o'clock	12H	PM 12 o'clock	32H
13 o'clock	13H	PM 01	21H
14 o'clock	14H	PM 02 o'clock	22H
15 o'clock	15H	PM 03 o'clock	23H
16 o'clock	16H	PM 04	24H
17 o'clock	17H	PM 05 o'clock	25H
18 o'clock	18H	PM 06 o'clock	26H
19 o'clock	19H	PM 07 o'clock	27H
20 o'clock	20H	PM 08	28H
21 o'clock	21H	PM 09 o'clock	29H
22 o'clock	22H	PM 10	30H
23 o'clock	23H	PM 11 o'clock	31H

13.4.6 day count register (RTC_DAY)

Address offset: 0x018

Reset value: indefinite

31	30	29	28	27	26	25	twenty	fourty	thirty	twenty	twenty	one	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	-
DAYH																
R/W																
DAYL																
R/W																

Bit	symbol	Function description
31:6	Reserved	-
5:4	DAYH	Day count tens value
3:0	DAYL	Day count single place value

The decimal system represents the day of 1:31, and the leap year and month are automatically calculated. The specific expression is as follows:

month	Day count indication
February (ordinary year)	01:28
February (leap year)	01:29
April, June, September, November	01:30
1, March, May, July, August, October, December	01:31

Page 351

13.4.7 Week count register (RTC_WEEK)

Address offset: 0x014

Reset value: indefinite

31	30	29	28	27	26	25	twent	twent	twent	twent	on	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WEEK R/W															

Bit	symbol	Function description
31:3	Reserved	-
2:0	WEEK	Week count value

Decimal 0:6 means Sunday:Saturday. Please write the correct decimal 0:6 BCD code, write other values, change

be ignored. The corresponding relationship of the weekly count value is as follows:
 week Week count indication

Sunday	00H
on Monday	01H
Tuesday	02H
Wednesday	03H
Thursday	04H
Friday	05H
Saturday	06H

Page 352**13.4.8 month count register (RTC_MON)**

Address offset: 0x01C

Reset value: indefinite

31	30	29	28	27	26	25	twent	twent	twent	twent	on	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MON															
R/W															

Bit	symbol	Function description
31:5	Reserved	-
4:0	MON	Monthly count value

Decimal 1:12 means 1:12 month. Please write the correct decimal 1:12 BCD code, write other values, and change
 be ignored.

13.4.9 years count register (RTC_YEAR)

Address offset: 0x020

Reset value: indefinite

31	30	29	28	27	26	25	twent	twent	twent	twent	on	19	18	17	16
Reserved															
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				Reserved					YEARH			YEARL			

Bit	symbol	Function description
31:8	Reserved	-
7:4	YEARH	Yearly count tens value
3:0	YEARL	Year's digit count value

The decimal 0:99 represents the year 0:99. Count according to the month round. Automatic calculation of leap years such as: 00, 04, 08,..., 92, 96 and so on. Please write the correct decimal year count value, write error value will be ignored.

Page 353**13.4.10 Minute Alarm Register (RTC_ALMMIN)**

Address offset: 0x024

Reset value: indefinite

31	30	29	28	27	26	25	twenty	forty	thirty	twenty	twenty	one	0	19	18	17	16
Reserved																	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
ALMMINH																	
Reserved																	

Bit	symbol	Function description
31:6	Reserved	-
5:4	ALMMINH	Ten digits of alarm matching value
3:0	ALMMINL	Minute alarm matching value ones place

Please set the BCD code of decimal 0:59. If other values are written, no alarm matching will occur.

13.4.11 hour alarm register (RTC_ALMHOUR)

Address offset: 0x028

Reset value: indefinite

31	30	29	28	27	26	25	twenty	forty	thirty	twenty	twenty	one	0	19	18	17	16
Reserved																	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
ALMHOURH																	
Reserved																	

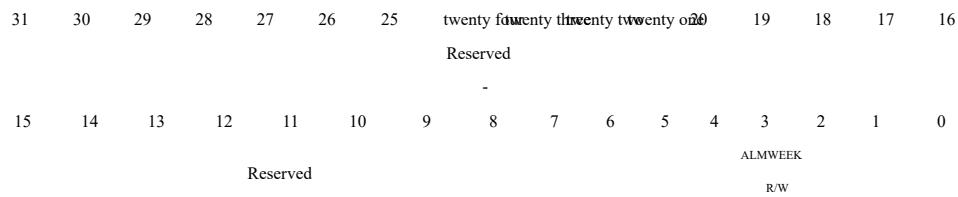
Bit	symbol	Function description
31:6	Reserved	-
5:4	ALMHOURH	Hour alarm ten-digit matching value
3:0	ALMHOURL	Match value of one place of the alarm clock

Please set the correct alarm matching value according to the time system, otherwise the time alarm matching will not occur.

Page 354**13.4.12 Weekly Alarm Register (RTC_ALMWEEK)**

Address offset: 0x02C

Reset value: indefinite



Bit symbol Function description

31:7 Reserved -

6:0 ALMWEEK Weekly alarm match value.

b0:b6 respectively correspond to Sunday:Saturday, corresponding to set "1", it means that the alarm is valid on that day of the week.
For example, b0=1, b5=1 means that the alarm setting on Sunday and Friday is valid.

Please set the correct alarm matching value according to the time system, otherwise the time alarm matching will not occur.

13.4.13 Clock Error Compensation Register (RTC_COMPEN)

Address offset: 0x030

*Only power-on reset is valid for this register, reset value: 0x00000020

Bit symbol Function description

31:16 Reserved-

15 EN Compensation enable

0: Disable clock error compensation

1. Euclidean sketch over communication

14:b Reserved-

8:0 CR Compensation values

By setting the compensation value, an accuracy compensation of +/-0.96ppm per second can be performed. The compensation value is 9 bits with small The 2's complement of the number point, the last 5 digits are the decimal part. The compensation range is 274.6ppm:212.6ppm. Minimum differential The error is +/-0.48ppm. The minimum resolution is 0.96ppm. Please refer to the table below for specific compensation accuracy:

Compensation value setting		Compensation number
EN	CR[8:0]	
1	1 0 0 0 0 0 0 0 -274.6ppm	
	1 0 0 0 0 0 0 0 1 -273.7ppm	
	
	0 0 0 0 1 1 1 1 1 -0.95ppm	
	0 0 0 1 0 0 0 0 0 0 ppm	
	
	0 1 1 1 1 1 1 0 +211.7ppm	
	0 1 1 1 1 1 1 1 +212.6ppm	
0	XXXXXXXXXX	no compensation

Explanation and calculation of compensation principle:

Since the counter uses a 32.768KHz clock to count, if you need to compensate for the accuracy per second, you can only press

According to the integer period compensation of 32.768KHz, the minimum unit of compensation per second is $(1/32768)*10^6 = 30.5\text{ppm}$,

Can not meet the requirements of high precision.

Then, to achieve high-precision clock compensation under the count clock of 32.768KHz, it is necessary to adjust the algorithm.

The compensation unit per second becomes $30.5\text{ppm}/32=0.96\text{ppm}$. Meet the high precision clock compensation requirements. And compensation occurs in a relatively uniform range every 32 seconds. Therefore, 5 decimal places are introduced into this register set up.

The set value is calculated as follows:

$$\text{CR[8:0]: } = \left(\frac{\text{Compensation target value} \times 10^{15}}{10^6} \right) + 0001.00000 \quad \text{B}$$

Take 2's complement

If the compensation target value is $+20.6\text{ppm}$, calculate the corresponding register value as follows:

$$\begin{aligned} \text{CR[8:0]} &= (20.3 \times 10^{15} / 10^6) \text{ take 2's complement} + 0001.00000 \text{B} \\ &= (0.6651904) \text{ Take 2's complement} + 0001.00000 \text{B} \\ &= 0000.10101 \text{B} + 0001.00000 \text{B} \\ &= 0001.10101 \text{B} \end{aligned}$$

If the compensation target value is -20.6ppm , calculate the corresponding register value as follows:

$$\begin{aligned} \text{CR[8:0]} &= (-20.3 \times 10^{15} / 10^6) \text{ 2's complement} + 0001.00000 \text{B} \\ &= (-0.6651904) \text{ Take 2's complement} + 0001.00000 \text{B} \\ &= 1111.01011 \text{B} + 0001.00000 \text{B} \\ &= 0000.01011 \text{B} \end{aligned}$$

14 Watchdog Timer (WDT)

14.1 Introduction to WDT

WDT can be used to detect and resolve faults caused by software errors. When the WDT counter reaches the set overflow time After that, it will trigger an interrupt or generate a system reset. The WDT is driven by a dedicated 10KHz on-chip oscillator.

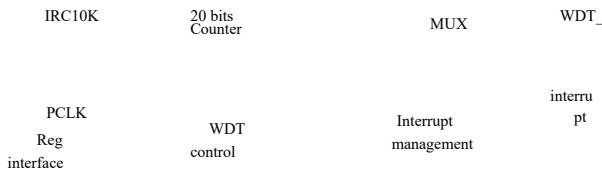


Figure 14-1 Overall block diagram of WDT

14.2 WDT function description

20Bit free-running incremental counter, the overflow time can be configured as 1.6ms-50s.

The action after overflow can be configured as interrupt or reset.

The WDT clock is provided by an independent RC oscillator and can work in Sleep and DeepSleep modes.

The WDTCON register can only be modified when the WDT is not started to prevent accidental modification after startup WDT configuration.

14.2.1 Generate an interrupt after WDT overflow

In this mode, WDT will periodically generate interrupts according to the set time. Need to be cleared in the interrupt service routine WDT interrupt flag.

The configuration method is as follows:

Step1: Configure WDT_CON. WOV and select WDT timeout time.

Step2: Set WDT_CON. WINT_EN to 1, choose to generate interrupt after WDT overflow.

Step3: Enable the WDT interrupt in the NVIC interrupt vector table.

Step4: Write 0x1E and 0xE1 to the WDT_RST register sequentially to start the WDT timer.

Step5: Write 0x1E and 0xE1 to the WDT_RST register in the interrupt service routine to clear the interrupt flag.

14.2.2 Reset after WDT overflow

In this mode, the Reset signal will be generated after the WDT counter overflows, which will reset the MCU. User program

It is necessary to clear the WDT counter before the WDT overflows to avoid WDT reset.

The configuration method is as follows:

Step1: Configure WDT_CON. WOV and select WDT counter overflow time.

Step2: Set WDT_CON. WINT_EN to 0, choose to generate reset after WDT overflow.

Step3: Write 0x1E and 0xE1 to the WDT_RST register to start the WDT timer.

Step4: Write 0x1E and 0xE1 to the WDT_RST register in sequence before the WDT overflows to clear the WDT count

Device.

Note: Since the WDT oscillator is a low-precision RC oscillator, it is strongly recommended that the WDT counter reaches the overflow

Clear the WDT before half of the value.

14.3 WDT register description

Base address 0X40000C00

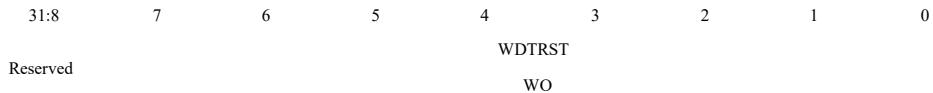
register	Offset address	describe
WDT_RST	0X080	WDT clear control register
WDT_CON	0X084	WDT control register

Table 14-1 List of WDT registers

14.3.1 WDT Clear Control Register (WDT_RST)

Offset address: 0x080

Reset value: 0x0000 0000



Bit symbol describe

31:8 Reserved Reserved bit, read as 0

7:0 WDTRST Watchdog start/clear control

When the watchdog is not started, write 0x1E, 0xE1 to this register in turn to start the WDT timer.

When the watchdog has started, write 0x1E and 0xE1 to this register in turn to clear the WDT timer and interrupt flag.

Chi.

Page 360

14.3.2 WDT_CON Register

Offset address: 0x084

Reset value: 0x0000 000F

Note:

- This register can be written only when the WDT is not running.

31:16	15:8	7	6	5	4	3	2	1	0
Reserved	WCNTL WDINT		Res.	WINT_EN	WDTR		WOV		
	RO	RO		R/W	RO			R/W	

Bit	symbol	describe												
31:16	Reserved	Reserved bit, read as 0												
15:8	WCNTL	The lower 8 bits of the WDT counter												
7	WDTINT	WDT interrupt flag <ul style="list-style-type: none"> 1: WDT interrupt has occurred, write 0x1E and 0xE1 to WDT_RST register to clear the interrupt flag. 0: No WDT interrupt occurred. 												
5	WINT_EN	Action configuration after WDT overflow <ul style="list-style-type: none"> 1: An interrupt is generated after the WDT overflows. 0: Reset after WDT overflow. 												
4	WDTR	WDT running flag <ul style="list-style-type: none"> 1: WDT is running 0: WDT stop 												
3:0	WOV[3:0]	WDT timer overflow time configuration <table border="0"> <tr> <td>0000: 1.6ms</td> <td>1000: 500ms</td> </tr> <tr> <td>0001: 3.2ms</td> <td>1001: 820ms</td> </tr> <tr> <td>0010: 6.4ms</td> <td>1010: 1.64s</td> </tr> <tr> <td>0011: 13ms</td> <td>1011: 3.28s</td> </tr> <tr> <td>0100: 26ms</td> <td>1100: 6.55s</td> </tr> <tr> <td>0101: 51ms</td> <td>1101: 13.1s</td> </tr> </table>	0000: 1.6ms	1000: 500ms	0001: 3.2ms	1001: 820ms	0010: 6.4ms	1010: 1.64s	0011: 13ms	1011: 3.28s	0100: 26ms	1100: 6.55s	0101: 51ms	1101: 13.1s
0000: 1.6ms	1000: 500ms													
0001: 3.2ms	1001: 820ms													
0010: 6.4ms	1010: 1.64s													
0011: 13ms	1011: 3.28s													
0100: 26ms	1100: 6.55s													
0101: 51ms	1101: 13.1s													

0110: 102ms	1110: 26.2s
0111: 205ms	1111: 52.4s

Page 361**15 Universal Synchronous Asynchronous Receiver Transmitter (UART)****15.1 Overview**

This product is equipped with 2 universal UART modules (UART0/1), the universal synchronous asynchronous receiver transmitter (UART) can be used for full-duplex data exchange with external devices, it supports synchronous one-way communication and multi-processor communication. Commonly used in short-distance, low-speed serial communication. The UART provides a variety of baud rates through a programmable baud rate generator. The baud rate of UART0 is generated by TIMER0, and the baud rate of UART1 is generated by TIMER1. UART supports multiple working modes.

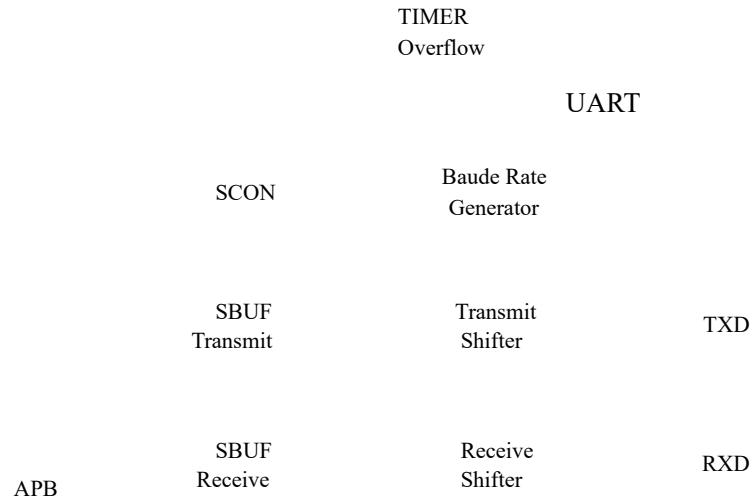
15.2 Block Diagram

Figure 15-1 UART structure block diagram

Page 362

15.3 Main features

The universal UART module supports the following basic functions:

Full duplex transmission, half duplex transmission

Programmable serial communication function

– Two character lengths: 8 bits, 9 bits

– Mode0/1/2/3 four transmission modes

16 bit baud rate generator

Multi-machine communication

Automatic address recognition

Page 363

15.4 Functional description

15.4.1 Working Mode

UART supports multiple working modes: synchronous half-duplex mode, asynchronous full-duplex mode. Via UARTx_SCON.SM0 With UARTx_SCON.SM1, you can configure various operating modes that you need.

15.4.1.1 Mode0~Mode3 function comparison

Configure UARTx_SCON.SM to choose different transmission modes: Mode0~Mode3. The master of these four working modes

The key function pairs are shown in the following table:

	Operating mode	Transmission bit width	Data composition	Baud rate
Mode0	Sync mode Half duplex	8bit Data(8bit)		= 12
Mode1	Asynchronous mode Full duplex	10bit Start (1bit) + Data(8bit) + Stop(1bit)		= (+ 1) $32 * (65536 -)$
Mode2	Asynchronous mode Full duplex	11bit Start (1bit) + Data(8bit) + B8(1bit) + Stop(1bit)		= (+ 1) 64
Mode3	Asynchronous mode Full duplex	11bit Start (1bit) + Data(8bit) + B8(1bit) + Stop(1bit)		= (+ 1) $32 * (65536 -)$

Table 15-1 Mode0/1/2/3 data structure

Note:

-Mode0 can only be used as a master to send UART synchronous shift clock, and cannot be used as a slave to receive external input UART synchronization

Shift clock.

- Represents the current PCLK frequency.

-Refer to UARTx_SCON for the definition of DBAUD.

-TM is the count value of TIMER. Note that TIMER must be configured as 16-bit auto-reload mode, counting register and reload

All registers have to be written with TM value.

15.4.1.2 Mode0 (synchronous mode, half-duplex)

When working in Mode0, the UART works in synchronous mode, and its baud rate is 1/12 of the fixed PCLK clock.

UART receiving data is input by RXD, UART sending data is output by TXD, and RXD is input and output at this time.

Out port. The UART synchronous shift clock is output by TXD, which is the output port at this time. Note that this mode only

It can be used as a master to send a synchronous shift clock, but it cannot be used as a slave to receive the clock from the outside. In this mode, the tran

The data bit width can only be 8 bits, there is no start bit and end bit.

Clear UARTx_SCON.SM0 and UARTx_SCON.SM1 to enter Mode0 working mode.

When sending data, clear the UARTx_SCON.REN bit and write the data to the UARTx_SBUF register. this

When the transmission data is output from RXD (low bit first, high bit last), the synchronous shift clock is output from TXD.

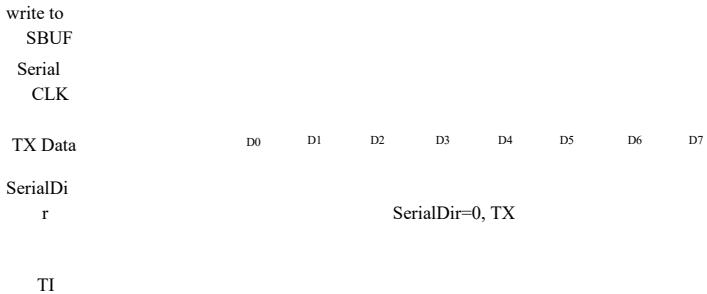


Figure 15-2 Mode0 sending data

When receiving data, set the `UARTx_SCON.REN` bit and clear the `UARTx_ISR.RI` bit. When receiving the knot
The data can be read from the `UARTx_SBUF` register. At this time, the received data is input from `RXD` (low order first,
The high bit is after), the synchronous shift clock is output from `TXD`.

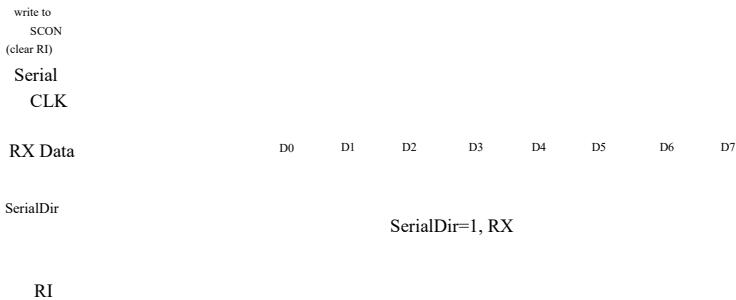


Figure 15-3 Mode0 receiving data

15.4.1.3 Mode1 (asynchronous mode, full duplex)

When working in Mode1, the sending data is sent through `TXD`, and the receiving data is received through `RXD`. The data

Consists of 10 bits: the start bit "0" starts, followed by 8 data bits (low bit first, high bit last), and finally the end

The beam position is "1".

In this mode, the baud rate is generated by the timer module and is programmable. The baud rate of `UART0` is determined by `TIMER0`
Generated, the baud rate of `UART1` is generated by `TIMER1`.

Clear `UARTx_SCON.SM0` to 0 and set `UARTx_SCON.SM1` to 1, to enter the Mode1 working mode.

When sending data, regardless of the value of `UARTx_SCON.REN`, write the sent data into the `UARTx_SBUF` register.

In the memory, the data will be shifted out from `TXD` (low bit first, high bit last).



TI

Figure 15-4 Model sending data

When receiving data, set the **UARTx_SCON.REN** bit to 1, and clear the **UARTx_ISR.RI** bit to 0. Start to pick up

Receive data on RXD (low bit first, high bit last), when the reception is complete, you can read the UARTx_SBUF register read out.

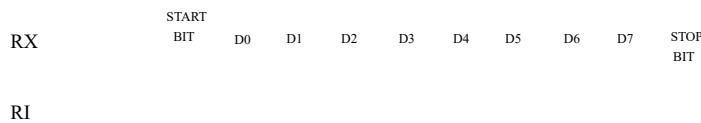


Figure 15-5 Model receiving data

15.4.1.4 Mode2 (asynchronous mode, full duplex)

When working in Mode2, the sending data is sent through TXD, and the receiving data is received through RXD. The data

Consists of 11 bits: the start bit "0" starts, followed by 8 data bits, 1 TB 8 bits and the end bit. additional

The TB8 bit is used in a multi-machine communication environment. When TB8=1, it indicates that the received address frame is; when TB8=0, indicates that the received is a data frame. When multi-machine communication is not required, this bit can also be used as a parity bit.

Page 366

In this mode, the baud rate can be generated independently, without external TIMER generation.

Set `UARTx_SCON.SM0` to 1 and clear `UARTx_SCON.SM1` to 0 to enter the Mode2 working mode.

When sending data, it has nothing to do with the value of `UARTx_SCON.REN`, and write the sent data into `UARTx_SBUF`

In the register, the data will be shifted out from TXD (low bit first, high bit last).

write to
SBUF



Figure 15-6 Mode2 sending data

When receiving data, set the `UARTx_SCON.REN` bit to 1, and clear the `UARTx_ISR.RI` bit to 0. Start to pick up

Receive data on RXD (low bit first, high bit last), when the reception is complete, you can read the UARTx_SBUF register read out.

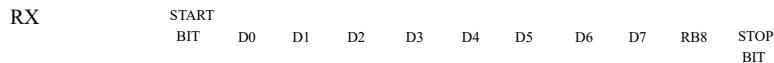


Figure 15-7 Mode2 receiving data

15.4.1.5 Mode3 (asynchronous mode, full duplex)

Mode3's data format, transmission timing and operation mode are the same as Mode2, the only difference is Mode3

The baud rate is generated by TIMER, rather than independently generated by the device itself like Mode2. Mode3 baud rate

It is programmable, and the baud rate generation method is the same as Mode1. In this product, the baud rate of UART0 is determined by TIMER0 is generated, and the baud rate of UART1 is generated by TIMER1.

Set UARTx_SCON.SM0 to 1, UARTx_SCON.SM1 to 1, to enter Mode3 working mode.

When sending data, it has nothing to do with the value of UARTx_SCON.REN, and write the sent data into UARTx_SBUF

In the register, the data will be shifted out from TXD (low bit first, high bit last).

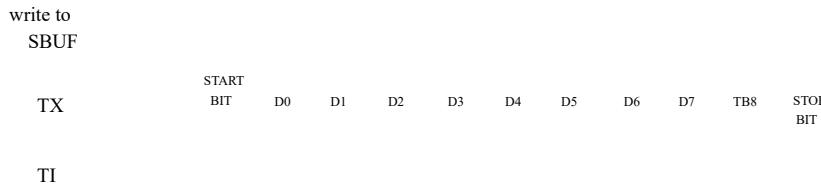


Figure 15-8 Mode3 sending data

When receiving data, set the UARTx_SCON.REN bit to 1, and clear the UARTx_ISR.RI bit to 0. Start to pick up

Receive data on RXD (low bit first, high bit last), when the reception is complete, you can read the UARTx_SBUF register read out.

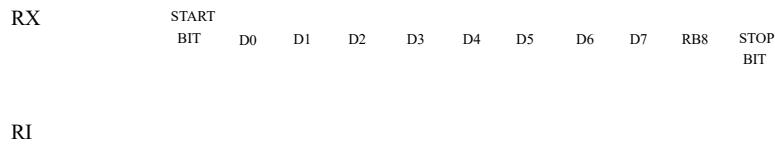


Figure 15-9 Mode3 receiving data

Page 368**15.4.2 Baud rate generation**

Mode0~Mode3 have different formulas for generating baud rate, see below for details:

Mode0 baud rate generation formula: =

₁₂

Mode1 baud rate generation formula: =

⁽⁺¹⁾
32*(65536-)

Mode2 baud rate generation formula: =

⁽⁺¹⁾
64

Mode3 baud rate generation formula: =

⁽⁺¹⁾
32*(65536-)

Note:

- Represents the current PCLK frequency.

-Refer to UARTx_SCON for the definition of DBAUD.

-TM is the count value of TIMER. Note that TIMER must be configured as 16-bit auto-reload mode, counting register and reload

All registers have to be written with TM value.

Page 369**15.4.2.1 Model/Mode3 baud rate setting example****PCLK = 1 MHz**

Baud rate	Dual baud rate		Single baud rate	
	CNT	Actual baud rate error %	CNT	Actual baud rate error %
2400	26	2403.85 0.16%	13	2403.85 0.16%
4800	13	4807.69 0.16%	7	4464.29 -6.99%
9600	7	8928.57 -6.99%	3	10416.67 8.51%
19200	3	20833.33 8.51%	2	15625.00 -18.62%
38400	2	31250.00 -18.62%	1	31250.00 -18.62%
57600	1	62500.00 8.51%	1	31250.00 -45.75%
76800	1	62500.00 -18.62%	0	#DIV/0! #DIV/0!
115200	1	62500.00 -45.75%	0	#DIV/0! #DIV/0!

PCLK = 4 MHz

Baud rate	Dual baud rate		Single baud rate	
	CNT	Actual baud rate error %	CNT	Actual baud rate error %
2400	104	2403.85 0.16%	52	2403.85 0.16%
4800	52	4807.69 0.16%	26	4807.69 0.16%
9600	26	9615.38 0.16%	13	9615.38 0.16%
19200	13	19230.77 0.16%	7	17857.14 -6.99%
38400	7	35714.29 -6.99%	3	41666.67 8.51%
57600	4	62500.00 8.51%	2	62500.00 8.51%
76800	3	83333.33 8.51%	2	62500.00 -18.62%
115200	2	125000.00 8.51%	1	125000.00 8.51%

PCLK = 10 MHz

Baud rate	Dual baud rate		Single baud rate	
	CNT	Actual baud rate error %	CNT	Actual baud rate error %
2400	260	2403.85 0.16%	130	2403.85 0.16%
4800	130	4807.69 0.16%	65	4807.69 0.16%
9600	65	9615.38 0.16%	33	9469.70 -1.36%
19200	33	18939.39 -1.36%	16	19531.25 1.73%
38400	16	39062.50 1.73%	8	39062.50 1.73%
57600	11	56818.18 -1.36%	5	62500.00 8.51%

Page 370

115200	5	125000.00 8.51%	3	104166.67 -9.58%
--------	---	-----------------	---	------------------

PCLK = 14 MHz

Dual baud rate			Single baud rate	
----------------	--	--	------------------	--

Baud rate	CNT	Actual baud rate error %	CNT	Actual baud rate error %
2400	365	2397.26 -0.11%	182	2403.85 0.16%
4800	182	4807.69 0.16%	91	4807.69 0.16%
9600	91	9615.38 0.16%	46	9510.87 -0.93%
19200	46	19021.74 -0.93%	twenty three	19021.74 -0.93%
38400	twenty three	38043.48 -0.93%	11	39772.73 3.57%
57600	15	58333.33 1.27%	8	54687.50 -5.06%
76800	11	79545.45 3.57%	6	72916.67 -5.06%
115200	8	109375.00 -5.06%	4	109375.00 -5.06%

PCLK = 20 MHz

Dual baud rate			Single baud rate	
----------------	--	--	------------------	--

Baud rate	CNT	Actual baud rate error %	CNT	Actual baud rate error %
2400	521	2399.23 -0.03%	260	2403.85 0.16%
4800	260	4807.69 0.16%	130	4807.69 0.16%
9600	130	9615.38 0.16%	65	9615.38 0.16%
19200	65	19230.77 0.16%	33	18939.39 -1.36%
38400	33	37878.79 -1.36%	16	39062.50 1.73%
57600	twenty two	56818.18 -1.36%	11	56818.18 -1.36%
76800	16	78125.00 1.73%	8	78125.00 1.73%
115200	11	113636.36 -1.36%	5	125000.00 8.51%

PCLK = 24 MHz

Dual baud rate			Single baud rate	
----------------	--	--	------------------	--

Baud rate	CNT	Actual baud rate error %	CNT	Actual baud rate error %
2400	625	2400.00 0.00%	313	2396.17 -0.16%
4800	313	4792.33 -0.16%	156	4807.69 0.16%
9600	156	9615.38 0.16%	78	9615.38 0.16%
19200	78	19230.77 0.16%	39	19230.77 0.16%
38400	39	38461.54 0.16%	20	37500.00 -2.34%

57600	26	57692.31 0.16%	13	57692.31 0.16%
76800	20	75000.00 -2.34%	10	75000.00 -2.34%
115200	13	115384.62 0.16%	7	107142.86 -6.99%

PCLK = 2 MHz

Dual baud rate			Single baud rate	
----------------	--	--	------------------	--

Baud rate	Actual baud rate error %		Actual baud rate error %	
-----------	--------------------------	--	--------------------------	--

	CNT₃₂		CNT₂₆	
2400	26	2403.85 0.16%	13	2403.85 0.16%
4800	26	4807.69 0.16%	13	4807.69 0.16%
9600	13	9615.38 0.16%	7	8928.57 -6.99%
19200	7	17857.14 -6.99%	3	20833.33 8.51%
38400	3	41666.67 8.51%	2	31250.00 -18.62%
57600	2	62500.00 8.51%	1	62500.00 8.51%
76800	2	62500.00 -18.62%	1	62500.00 -18.62%
115200	1	125000.00 8.51%	1	62500.00 -45.75%

PCLK = 8 MHz

Baud rate	Dual baud rate		Single baud rate	
	CNT	Actual baud rate error %	CNT	Actual baud rate error %
2400	208	2403.85 0.16%	104	2403.85 0.16%
4800	104	4807.69 0.16%	52	4807.69 0.16%
9600	52	9615.38 0.16%	26	9615.38 0.16%
19200	26	19230.77 0.16%	13	19230.77 0.16%
38400	13	38461.54 0.16%	7	35714.29 -6.99%
57600	9	55555.56 -3.55%	4	62500.00 8.51%
76800	7	71428.57 -6.99%	3	83333.33 8.51%
115200	4	125000.00 8.51%	2	125000.00 8.51%

PCLK = 11.0592 MHz

Baud rate	Dual baud rate		Single baud rate	
	CNT	Actual baud rate error %	CNT	Actual baud rate error %
2400	288	2400.00 0.00%	144	2400.00 0.00%
4800	144	4800.00 0.00%	72	4800.00 0.00%
9600	72	9600.00 0.00%	36	9600.00 0.00%

19200	36	19200.00 0.00%	18	19200.00 0.00%
38400	18	38400.00 0.00%	9	38400.00 0.00%
57600	12	57600.00 0.00%	6	57600.00 0.00%
76800	9	76800.00 0.00%	5	69120.00 -10.00%
115200	6	115200.00 0.00%	3	115200.00 0.00%

PCLK = 16 MHz

Baud rate	Dual baud rate		Single baud rate	
	CNT	Actual baud rate error %	CNT	Actual baud rate error %
2400	417	2398.08 -0.08%	208	2403.85 0.16%
4800	208	4807.69 0.16%	104	4807.69 0.16%
9600	104	9615.38 0.16%	52	9615.38 0.16%
19200	52	19230.77 0.16%	26	19230.77 0.16%
38400	26	38461.54 0.16%	13	38461.54 0.16%
57600	17	58823.53 2.12%	9	55555.56 -3.55%

76800	13	76923.08 0.16%	7	71,428.57 -6.99%
115200	9	111111.11 -3.55%	4	125000.00 8.51%

PCLK = 32 MHz

Baud rate	Dual baud rate		Single baud rate	
	CNT	Actual baud rate error %	CNT	Actual baud rate error %
2400	833	2400.96 0.04%	417	2398.08 -0.08%
4800	417	4796.16 -0.08%	208	4807.69 0.16%
9600	208	9615.38 0.16%	104	9615.38 0.16%
19200	104	19230.77 0.16%	52	19230.77 0.16%
38400	52	38461.54 0.16%	26	38461.54 0.16%
57600	35	57142.86 -0.79%	17	58823.53 2.12%
76800	26	76923.08 0.16%	13	76923.08 0.16%
115200	17	117647.06 2.12%	9	111111.11 -3.55%

PCLK = 22.12 MHz

Baud rate	Dual baud rate		Single baud rate	
	CNT	Actual baud rate error %	CNT	Actual baud rate error %
2400	576	2400.17 0.01%	288	2400.17 0.01%
4800	288	4800.35 0.01%	144	4800.35 0.01%

Page 373

9600	144	9600.69 0.01%	72	9600.69 0.01%
19200	72	19201.39 0.01%	36	19201.39 0.01%
38400	36	38402.78 0.01%	18	38402.78 0.01%
57600	twenty four	57604.17 0.01%	12	57604.17 0.01%
76800	18	76805.56 0.01%	9	76805.56 0.01%
115200	12	115208.33 0.01%	6	115208.33 0.01%

15.5 Frame error detection

When working in Mode1/2/3, the UART has a frame error detection function, and the hardware will automatically detect the number of frames received. According to whether there is a valid Stop bit. If the received data is that the hardware did not receive a valid Stop bit as expected, If synchronization fails or excessive noise is present, UARTx_ISR.FE is set to 1. The UARTx_ISR.FE bit is set by hardware. The software is cleared to 0. If the software does not clear it to 0 in time, the subsequent received data will not be cleared even if it has a valid Stop bit. The UARTx_ISR.FE flag is cleared to 0.

Page 374

15.6 Multi-machine communication

Mode2/3 has multi-machine communication function, for this reason, 1 bit TB8/RB8 is added to its frame format. will
UARTx_SCON.SM2 is set to "1" to enable the multi-machine communication bit.

When the multi-machine communication bit is turned on, when sending data, the host can distinguish the current frame through UARTx_SCON.TB8
Is it an address frame (UARTx_SCON.TB8=1) or a data frame (UARTx_SCON.TB8=0). When receiving data,
The slave will ignore the current received frame whose RB8 bit (bit 9) is "0".

- When it is a data frame, the frame data will not be stored in the UARTx_SBUF register of the slave, and the slave will not be receiving
Off.
- When it is an address frame, since the automatic address recognition function in the multi-machine communication has been turned on, the slave machine can detect the received addr
Whether its own address matches.
 - If the addresses match, the slave will set UARTx_SCON.RB8 to "1" and UARTx_ISR.RI to "1". Slave software
After seeing UARTx_SCON.RB8=1 and UARTx_ISR.RI=1, clear the UARTx_SCON.SM2 bit to "0",
Accept data frames.
 - If the address does not match, it indicates that the master is not addressing the slave, and the slave hardware keeps UARTx_SCON.RB8 and
UARTx_ISR.RI is "0", and the software keeps the UARTx_SCON.SM2 bit as "1" and continues to be in the address monitoring state.

Note: If necessary, the multi-machine communication bit can also be turned on under Mode1, at this time the TB8 bit is replaced by the stop bit.
When the slave receives a matching address frame and a valid stop bit, UARTx_ISR.RI will be set to "1".

Page 375

15.7 Automatic address recognition

When the multi-machine communication bit is turned on (UARTx_SCON.SM2 is set to "1"), the automatic address recognition function will also be turned on. It can be implemented by hardware, so that the slave can detect and receive each address frame, if the address matches the address of the slave, The receiving end will give UARTx_ISR.RI receiving flag. If the address does not match, the receiving end will not give any connection Close sign.

15.7.1 Given address

The UARTx_SADDR register of the UART device is used to represent the given address of its own device,

The UARTx_SADEF register is an address mask, which can be used to define extraneous bits in the address. when

A bit of UARTx_SADEF is "0", which means that the address of this bit is a don't care bit, that is to say in the address matching process

, This bit address does not participate in address matching. These extraneous bits increase the flexibility of addressing, so that the host can simultaneously address one or more slave devices. Note that if you need to give a unique matching address, send UARTx_SADEF

The register must be set to 8'hFF.

GivenAddr = SADDR & SADEF

15.7.1.1 Broadcast address

The broadcast address is used to address all slave devices at the same time, and the general broadcast address is 8'hFF.

BroadCastA ddr = SADDR | SADEF

15.7.1.2 Examples

Suppose the UARTx_SADDR and UARTx_SADEF configuration of a slave machine are as follows:

SADDR: 8'b01101001

SADEF: 8'b11111011

Then its given address and broadcast address are as follows:

Given: 8'b01101x01

Broadcast: 8'b11111x11

It can be seen that the host can use four addresses to address the slave, which are:

8'b01101001 and 8'b01101101 (given address)

8'b11111011 and 8'b11111111 (broadcast address).

Page 376

15.8 Transceiver end buffer

15.8.1 Receive Cache

The universal UART (UART0/1) receiver has a frame length (8/9bits) receiving buffer, which means that when a frame

After the data is received, the data in the receive buffer will be kept until the Stop bit of the next frame of data is received

After finishing, the receiving buffer will be updated to a new frame of data.

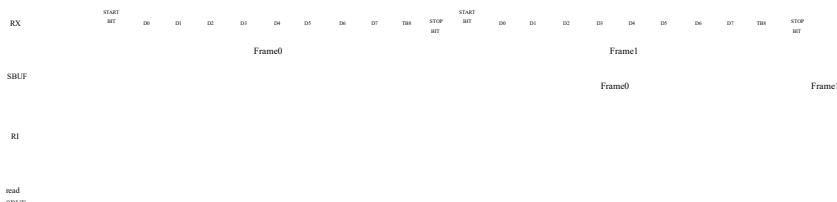


Figure 15-10 Receive buffer

15.8.2 Send buffer

The general UART (UART0/1) transmitter does not support transmit buffer. If in the process of sending data, fill in

The UARTx_SBUF register will destroy the data currently being sent. Software should avoid this kind of operation.

Page 377

15.9 Register

UART0 base address: 0x4000 0000

UART1 base address: 0x4000 0100

register	Offset address	describe
UARTx_SBUF	0x00	Data register
UARTx_SCON	0x04	Control register
UARTx_SADDR	0x08	Address register
UARTx_SADEN	0x0C	Address mask register
UARTx_ISR	0x10	Interrupt flag register
UARTx_ICR	0x14	Interrupt flag clear register

15.9.1 Data Register (**UARTx_SBUF**)

Offset address: 0x00

Reset value: 0x0000 0000

Bit	mark	Function description
31:8	Reserved	
7:0	SBUF	When sending data, write the data to be sent into this register; When receiving data, read the received data from this register;

Note that the value read from this register is actually the value in RxBuffer, and the value written to this register is actually written to TXShifter.

15.9.2 Control Register (**UARTx_SCON**)

Offset address: 0x04

Reset value: 0x0000 0000

User manual

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Reserved				DBA	Reser	SM01	SM2	REN	TB8	RB8	TIEN	RIEN	
						UD	ved								
		R				RW	R	RW	RW	RW	RW	RW	RW	RW	RW

Bit	mark	Function description
31:10	Reserved	
9	DBAUD	Baud rate multiplier setting 0: Single baud rate; 1: Double baud rate
8	Reserved	
7:6	SM01	Working mode configuration 00: mode0; 01: mode1; 10: mode2; 11: mode3
5	SM2	Multi-machine communication enable control 0: Turn off the multi-machine communication function 1: Enable multi-machine communication function
4	REN	Receive enable control Mode0: 0: send, 1: receive Others: 0: send, 1: receive/send
3	TB8	TB8 bits to be sent when sending data
2	RB8	RB8 bits received when receiving data
1	TIEN	Send complete interrupt enable 0: Disable sending completion interrupt 1: Enable send completion interrupt
0	RIEN	Receive complete interrupt enable 0: Disable receiving completion interrupt 1: Enable receiving completion interrupt

15.9.3 Address Register (UARTx_SADDR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16
							Reserved							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
		Reserved								SADDR				

Bit	mark	Function description
31:8	Reserved	
7:0	SADDR	Slave device address

15.9.4 Address Mask Register (UARTx_SADEN)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
SADEN															
RW															
Bit	mark														
31:8	Reserved														
7:0	Slave device address mask														

Page 380**15.9.5 Flag bit register (UARTx_ISR)**

Offset address: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
FE TI RI															
R R R															

Bit	symbol	describe
31:3	Reserved	
2	FE	Receive frame error flag; set by hardware, cleared by software 1: FE interrupt flag is valid 0: FE interrupt flag is invalid
1	TI	Send completion interrupt flag bit; set by hardware, cleared by software 1: TI interrupt flag is valid 0: TI interrupt flag is invalid
0	RI	Receive complete interrupt flag bit; set by hardware, cleared by software 1: RI interrupt flag is valid 0: RI interrupt flag is invalid

Page 381**15.9.6 Flag bit clear register (**UARTx_ICR**)**

Offset address: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four	twenty five	twenty six	twenty seven	twenty eight	on	20	19	18	17	16
Reserved																	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Reserved																	
														FE	TI	RI	
														CLR	CLR	CLR	
														W	W	W	

Bit	mark	Function description
31:3 RESERVED		
2	FECLR	Receiving frame error flag clear Write 0 to clear zero, write 1 invalid
1	TICLR	Send complete interrupt flag clear Write 0 to clear zero, write 1 invalid
0	RICLR	Reception complete interrupt flag clear Write 0 to clear zero, write 1 invalid

Page 382

16 Low Power Synchronous Asynchronous Receiver Transmitter (LPUART)

16.1 Overview

This product is equipped with 1 LPUART module, LPUART includes all necessary hardware support, so that the minimum power consumption Asynchronous serial communication can be carried out under the following conditions; the baud rate can be generated by the external TIMER2 or by the system timer.

In order to support low-power applications, LPUART adds a SCLK clock in addition to the original PCLK clock.

The internal register configuration logic of the LPUART module works in the PCLK clock domain, and the data sending and receiving logic works in the SCLK clock domain. When the system enters the low-power mode, turn off the high-frequency PCLK clock and turn on the low-frequency SCLK clock. LPUART can still send and receive data normally.

SCLK clock source can be selected: PCLK, external low-speed clock (XTL), internal low-speed clock (RCL). exist When LPMODE=1, SCLK clock also supports 1/2/4/8/16/32/64/128 times prescaler.

Note that when LPMODE=0, LPUART receives the Toggle output signal of the TIMER2 clock instead of Overflow signal, so the Toggle output of TIMER2 must be enabled.

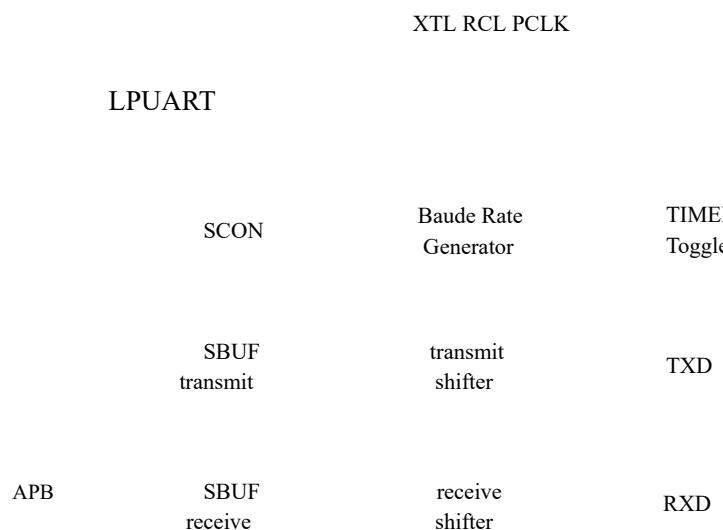
Page 383**16.2 Block Diagram**

Figure 16-1 LPUART structure block diagram

Page 384

16.3 Main features

The LPUART module supports the following basic functions:

Configure clock PCLK

Transmission clock SCLK (SCLK can choose XTL, RCL and PCLK)

Send and receive data in low power consumption mode of the system

Full duplex transmission, half duplex transmission

Programmable serial communication function

– Two character lengths: 8 bits, 9 bits

– Mode0/1/2/3 four transmission modes

16 bit baud rate counter

Multi-machine communication

Automatic address recognition

16.4 Functional description

16.4.1 Configuration Clock and Transmission Clock

The LPUART module has two clocks: configuration clock PCLK and transmission clock SCLK.

Configure clock

The configuration clock is used to configure the registers of the LPUART module on the system APB bus and is fixed to PCLK.

Transmission clock

The transmission clock is used for the LPUART data receiving and sending logic, and XTL, RCL and PCLK can be selected. When SCLK

It is XTL or RCL. When the system enters DeepSleep, LPUART can still send and receive data normally.

16.4.2 Working Mode

LPUART supports multiple working modes: synchronous half-duplex mode, asynchronous full-duplex mode. By configuration

The value of LPUARTx_SCON.SM can obtain the required working mode.

LPUART has an LPMODE control bit added to UART. When the position is "1", only Mode1/3 is supported

The working mode and the way of generating the baud rate will also change.

16.4.2.1 Mode0~Mode3 function comparison

When **LPMODE=0** :

Configure LPUARTx_SCON.SM to select 4 transmission modes, the function comparison of each mode is as follows:

	Operating mode	Transmission bit width	Data composition	Baud rate
Mode0	Sync mode			
	Half duplex	8bit	Data(8bit)	= 12
Mode1	Asynchronous mode		Start (1bit) + Data(8bit)	(+ 1)
	Full duplex	10bit	+ Stop(1bit)	$32 * (65536 -)$
Mode2	Asynchronous mode		Start (1bit) + Data(8bit)	(+ 1)
	Full duplex	11bit	+ B8(1bit) + Stop(1bit)	64
	Asynchronous mode		Start (1bit) + Data(8bit)	

Mode3	11bit	=	$\frac{1}{32} * (65536 -)$
Full duplex	+ B8(1bit) + Stop(1bit)		

Table 16-1 Mode0/1/2/3 Data Structure

When LPMODE=1:

Mode2	Asynchronous mode Full duplex	11bit + B8(1bit) + Stop(1bit)	=	$* 4$
-------	----------------------------------	----------------------------------	---	-------

Note:

- Mode0 can only be used as a master to send LPUART synchronous shift clock, and cannot be used as a slave to receive external input LPUART synchronous shift clock.
- Represents the current SCLK frequency.
- Refer to LPUARTx_SCON for the definition of DBAUD.
- TM is the TIMER count value. Note that TIMER must be configured in 16-bit auto-reload mode, counting register and reload. The TM value must be written into the load register.
- PreScale is the prescaler coefficient.

16.4.2.2 Mode0 (synchronous mode, half-duplex) data transmission and reception instructions

Set LPUART_SCON.SM to 0, LpUart works in Mode0, clock and data synchronous input and output, wave

The special rate is fixed at SCLK/12. The width of the transmitted data is fixed at 8 bits, and there is no start bit and end bit. data

Both receiving and sending are realized through the RXD pin; the synchronous shift clock is output by the TXD pin. Note that TXD cited Pin does not accept input clock.

When LPMODE=1, Mode0 working mode is not supported.

When sending data, set LPUART_SCON.REN to 0 and write the data to be sent into the SBUF register.

At this time, the transmission data will be output from RXD (low bit first, high bit last), and the synchronous shift clock will be output from TXD.



Figure 16-2 Mode0 sending data

When receiving data, set LPUART_SCON.REN to 1, and clear the LPUART_ISR.RI bit. When picking up

After receiving, the data can be read from the LPUART_SBUF register. At this time, the received data is input from RXD (low bit first, high bit last), the synchronous shift clock is output from TXD.

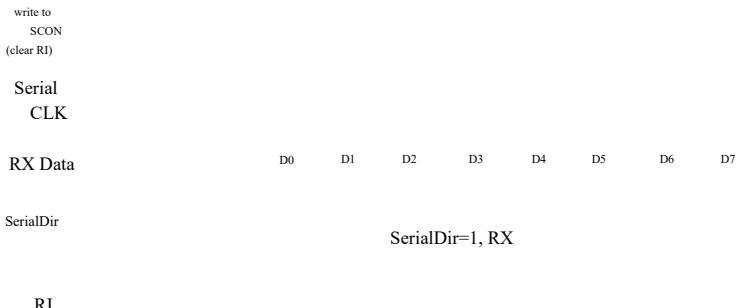


Figure 16-3 Mode0 receiving data

16.4.2.3 Mode1 (asynchronous mode, full-duplex) data sending and receiving instructions

When working in Mode1, the sending data is sent through TXD, and the receiving data is received through RXD. The data

Consists of 10 bits: the start bit "0" starts, followed by 8 data bits (low bit first, high bit last), and finally the end

The beam position is "1".

In this mode, the baud rate of LPUART is generated by the timer TIMER2 module and is programmable.

Set LPUART_SCON.SM0 to 0 and LPUART_SCON.SM1 to 1, to enter the Mode1 working mode.

When LPMODE=1, Mode1 working mode is supported, but the baud rate calculation method is changed, the specific parameters

Check the baud rate generation chapter.

When sending data, regardless of the value of LPUART_SCON.REN, write the sent data into LPUART_SBUF

In the register, the data will be shifted out from TXD (low bit first, high bit last).

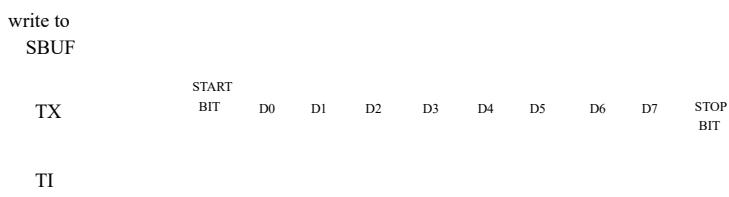


Figure 16-4 Mode1 sending data

When receiving data, you need to set the LPUART_SCON.REN bit and clear the LPUART_ISR.RI bit to 0. open

Start receiving the data on RXD (low bit first, high bit last), when the reception is complete, you can read the data from LPUART_SBUF

Register read.



RI

Figure 16-5 Model receiving data

Page 389**16.4.2.4 Mode2** (asynchronous mode, full-duplex) data transmission and reception instructions

When working in Mode2, the sending data is sent through TXD, and the receiving data is received through RXD. The data

Consists of 11 bits: the start bit "0" starts, followed by 8 data bits, 1 TB 8 bits and the end bit. additional

The TB8 bit is used in a multi-machine communication environment. When TB8=1, it indicates that the received address frame is; when TB8=0,

Indicates that the received is a data frame. When multi-machine communication is not required, this bit can also be used as a parity bit.

In this mode, the baud rate can be generated independently, without external TIMER generation.

Set LPUART_SCON.SM0 to 1 and clear LPUART_SCON.SM1 to 0 to enter Mode2 working mode.

When LPMODE=1, Mode2 working mode is not supported.

When sending data, it has nothing to do with the value of LPUART_SCON.REN, and write the sent data

In the LPUART_SBUF register, the data will be shifted out from TXD (low bit first, high bit last).

write to
SBUF



TI

Figure 16-6 Mode2 sending data

When receiving data, you need to set the LPUART_SCON.REN bit and clear the LPUART_ISR.RI bit to 0. open

Start receiving the data on RXD (low bit first, high bit last), when the reception is complete, you can read the data from LPUART_SBUF

Register read.



RI

Figure 16-7 Mode2 receiving data

Page 390**16.4.2.5 Mode3** (asynchronous mode, full-duplex) data transmission and reception instructions

Mode3's data format, transmission timing and operation mode are the same as Mode2, the only difference is Mode3

The baud rate is generated by TIMER, rather than independently generated by the device itself like Mode2. Mode3 baud rate

It is programmable, and the baud rate generation method is the same as Mode1.

Set LPUART_SCON.SM0 to 1, LPUART_SCON.SM1 to 1, to enter Mode3 working mode.

When LPMODE=1, Mode3 working mode is supported. But the baud rate calculation method has changed, the specific parameters

Check the baud rate generation chapter.

When sending data, it has nothing to do with the value of LPUART_SCON.REN, and write the sent data

In the LPUART_SBUF register, the data will be shifted out from TXD (low bit first, high bit last).

write to
SBUF

TX	START BIT D0 D1 D2 D3 D4 D5 D6 D7 TB8 STOP BIT
----	--

TI

Figure 16-8 Mode3 sending data

When receiving data, you need to set the LPUART_SCON.REN bit and clear the LPUART_ISR.RI bit to 0. open

Start receiving the data on RXD (low bit first, high bit last), when the reception is complete, you can read the data from LPUART_SBUF Register read.

RX	START BIT D0 D1 D2 D3 D4 D5 D6 D7 RB8 STOP BIT
----	--

RI

Figure 16-9 Mode3 receiving data

Page 391

16.4.3 Baud rate generation

LPMODE=0

The formulas of Mode0~Mode3 to generate baud rate are as follows:

Mode0 baud rate generation formula: =

¹²

Mode1 baud rate generation formula: =

⁽⁺¹⁾
32*(65536-)

Mode2 baud rate generation formula: =

⁽⁺¹⁾
64

Mode3 baud rate generation formula: =

⁽⁺¹⁾
32*(65536-)

LPMODE=1

When LPMODE=1, Mode0 and Mode2 are not supported.

Mode1, Mode3 baud rate generation formula: =

^{*4}

Note:

- Represents the current SCLK frequency.
- Refer to LPUARTx_SCON for the definition of DBAUD.
- TM is the TIMER count value. Note that TIMER must be configured in 16-bit auto-reload mode, counting register and reload. The TM value must be written into the load register.
- PreScale is the prescaler coefficient.

16.5 Frame Error Detection

When working in Model1/2/3, LPUART has a frame error detection function, and the hardware will automatically detect the received frame

Whether the data has a valid Stop bit. If you do not receive a valid Stop bit within the expected time when receiving data,

If synchronization failure or excessive noise occurs, the LPUART_ISR.FE bit is set to 1. The LPUART_ISR.FE bit is hard

The software is set to 1, and the software is cleared to 0. If the software is not cleared to 0 in time, the subsequent received data will not be

The LPUART_ISR.FE flag will be cleared to 0.

16.6 Multi-machine communication

Mode2/3 has multi-machine communication function, for this reason, 1 bit TB8/RB8 is added to its frame format. SCON.SM2

Set "1" to enable the multi-machine communication bit.

When the multi-machine communication bit is turned on, when sending data, the host can use LPUART_SCON.TB8 to distinguish the current

Is the frame an address frame (LPUART_SCON.TB8=1) or a data frame (LPUART_SCON.TB8=0).

- When it is a data frame, the frame data will not be stored in the LPUARTx_SBUF register of the slave, and the slave will not receive interrupted.
- When it is an address frame, since the automatic address recognition function in the multi-machine communication has been turned on, the slave machine can detect the received address whether its own address matches.
 - If the addresses match, the slave will set LPUARTx_ISR.RI to "1" and LPUARTx_SCON.RB to "1". Slave

Software see	LPUARTx_SCON.RB8=1	and	LPUARTx_ISR.RI=1	After
--------------	--------------------	-----	------------------	-------

The LPUARTx_SCON.SM2 bit is cleared to "0" to accept the data frame.

 - If the address does not match, it indicates that the master is not addressing the slave, and the slave hardware keeps LPUARTx_RB8 and LPUARTx_ISR.RI are "0", the software keeps the LPUARTx_SCON.SM2 bit as "1", and the slave continues to be in the address monitoring state.

Note:

- If necessary, you can also turn on the multi-machine communication bit in Mode1, at this time the TB8 bit is replaced by the stop bit. When the slave is connected when a matching address frame and a valid stop bit are received, LPUARTx_ISR.RC will be set to "1".

16.7 Automatic address recognition

When the multi-machine communication bit is turned on (LPUART_SCON.SM2 is set to "1"), the automatic address recognition function will also be implemented. The function is implemented by hardware, so that the slave can detect each address frame received, if the address matches the address of the slave, the receiving end will give LPUART_ISR.RI receiving flag. If the address does not match, the receiving end will not give any receive logo.

16.7.1 Given address

The LPUART_SADDR register of the LPUART device is used to represent the given address of its own device, The LPUART_SADEN register is an address mask, which can be used to define extraneous bits in the address. when A bit of LPUART_SADEN is "0", indicating that the address of this bit is a don't care bit, that is to say, during the address matching process

, This bit address does not participate in address matching. These extraneous bits increase the flexibility of addressing, so that the host can simultaneously address one or more slave devices. Note that if you need to give a unique matching address, send LPUART_SADEN

The register must be set to 8'hFF.

GivenAddr = SADDR & SADEN

16.7.2 Broadcast address

The broadcast address is used to address all slave devices at the same time, and the general broadcast address is 8'hFF.

BroadCastA ddr = SADDR | SADEN

16.7.3 Examples

Suppose the configuration of LPUART_SADDR and LPUART_SADEN of a slave is as follows:

SADDR: 8'b01101001

SADEN: 8'b11111011

Then its given address and broadcast address are as follows:

Given: 8'b01101x01

Broadcast: 8'b11111x11

It can be seen that the host can use four addresses to address the slave, which are:

8'b01101001 and 8'b01101101 (given address)

8'b11111011 and 8'b11111111 (broadcast address).

16.8 Transceiver end buffer

16.8.1 Receive Cache

The LPUART receiver has a frame length (8/9bits) receive buffer, which means that when a frame of data is received

After that, the data in the receive buffer will be kept until the Stop bit of the next frame of data is received.

The cache will be updated with a new frame of data.

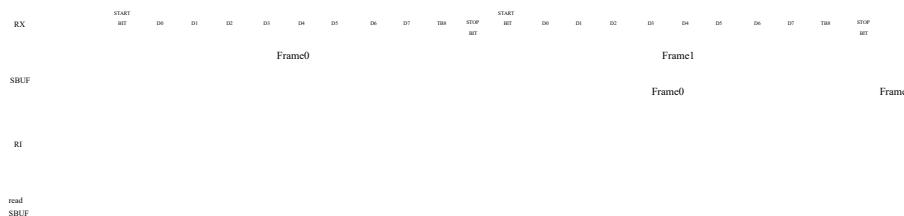


Figure 16-10 Receive buffer

16.8.2 Send buffer

The LPUART transmitter has a frame length (8/9bits) transmission buffer. When the transmission shift register is transmitting the current

For frame data, the CPU can write the data to be sent in the next frame to the sending buffer.

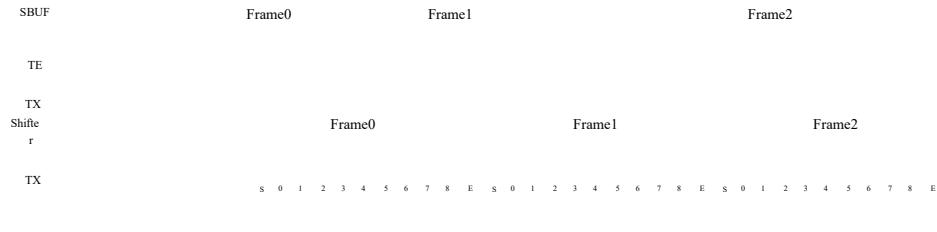


Figure 16-11 Send buffer

Among them, the register bit LPUART_ISR.TE is the transmit buffer empty and full flag bit. The LPUART module contains only one frame (8/9bits) transmit buffer, so when the LPUART_ISR.TE bit is "1", the hardware will automatically mask the software pair. The LPUART_SBUF register is written until the LPUART_ISR.TE bit becomes "0". Software will be released. Before sending data into the LPUART_SBUF register, you must determine the status of the LPUART_ISR.TE bit "0" and "1". Otherwise, the sending data will be lost.

Page 395

16.9 Register

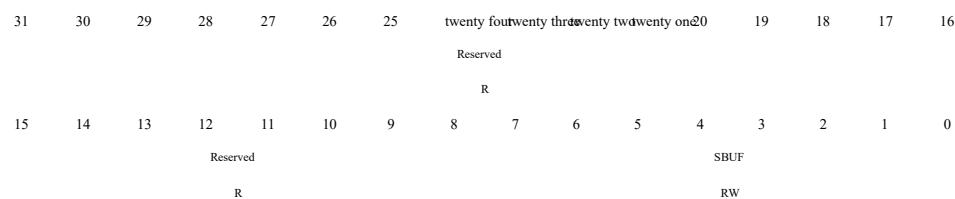
LPUART base address: 0x4000 0200

Register name	Offset address	describe
LPUART_SBUF	0x00	Data register
LPUART_SCON	0x04	Control register
LPUART_SADDR	0x08	Address register
LPUART_SADEN	0x0C	Address mask register
LPUART_ISR	0x10	Interrupt flag register
LPUART_ICR	0x14	Interrupt flag clear register

16.9.1 Data Register (LPUART_SBUF)

Offset address: 0x00

Reset value: 0x0000 0000



Bit	mark	Function description
31:8	Reserved	
7:0	SBUF	When sending data, when sending data is written into this register; when receiving data, after the data is received, it is read from this register.

Note that the value read from this register is actually the value in RXBuffer, and the value written to this register is actually written to TXShifter.

Page 396**16.9.2 Control Register (LPUART_SCON)**

Offset address: 0x04

Reset value: 0x0000 E000

31	30	29	28	27	26	25	twentynine	twentynine	threentwenty	twentynine	twentynine	on20	19	18	17	16
Reserved																
							R									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	PRS		SCLKSEL		LPM	DBA	TEEN		SM01	SM2	REN	TB8	RB8	TIEN	RIEN	
					ODE		UD									
	RW		RW						RW		RW		RW		RW	

Bit	mark	Function description
31:16	Reserved	
15:13	PRS	Transmission clock SCLK prescaler selection; 000: div128; 001: div64; 010: div32; ... ;110:div2;111:div1 PRS[2:0] is only valid when LPMODE=1; when LPMODE=0, PRS[2:0] will not prescale SCLK.
12:11	SCLKSEL	Transmission clock SCLK selection; 00: PCLK; 01: PCLK; 10: XTL; 11: RCL
10	LPMODE	Low power consumption mode; 0: normal working mode; 1: Low power consumption working mode
9	DBAUD	Double baud rate; 0: Single baud rate; 1: Double baud rate
8	TEEN	Send buffer empty interrupt enable; 0: disable; 1: enable
7:6	SM01	Operating mode; 00: mode0; 01: mode1; 10: mode2; 11: mode3
5	SM2	Multi-host communication; 0: disable, 1: enable

Page 397

4	REN	Receive enable; mode0: 0: send, 1: receive Others: 0: send, 1: receive/send
3	TB8	Send TB8 bits
2	RB8	Receive RB8 bit
1	TIEN	Send completion interrupt enable; 0: disable; 1: enable
0	RIEN	Receive completion interrupt enable; 0: disable; 1: enable

Page 398

16.9.3 Address Register (LPUART_SADDR)

Offset address: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four twenty three twenty two twenty one	20	19	18	17	16	
Reserved													
15	14	13	12	11	10	9	8	7	6	5	4	3	2
R													
15	14	13	12	11	10	9	8	7	6	5	4	3	2
SADDR													
RW													

Bit	mark	Function description
31:8	Reserved	
7:0	SADDR	Slave device address register

16.9.4 Address Mask Register (LPUART_SADEN)

Offset address: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four twenty three twenty two twenty one	20	19	18	17	16	
Reserved													
15	14	13	12	11	10	9	8	7	6	5	4	3	2
R													
15	14	13	12	11	10	9	8	7	6	5	4	3	2
SADEN													
RW													

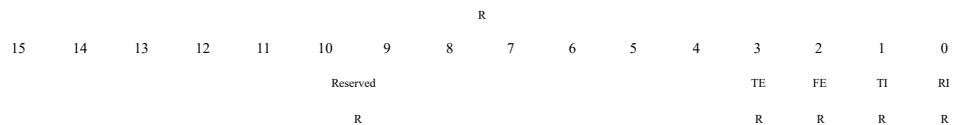
Bit	symbol	Function description
31:8	Reserved	
7:0	SADEN	Slave device address mask register

16.9.5 Interrupt Flag Bit Register (LPUART_ISR)

Offset address: 0x10

Reset value: 0x0000 0008

31	30	29	28	27	26	25	twenty four twenty three twenty two twenty one	20	19	18	17	16	
Reserved													



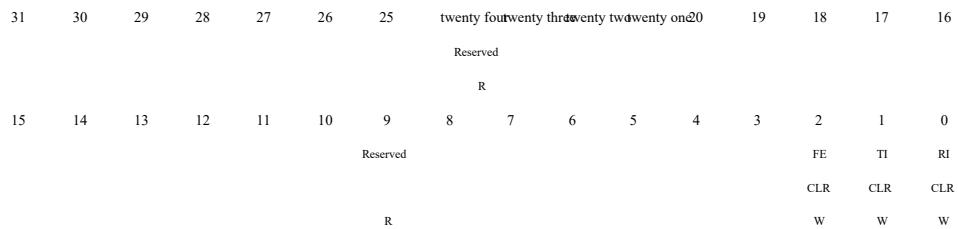
Bit	mark	Function description
31:4	Reserved	
3	TE	Transmit buffer empty interrupt flag bit, set by hardware, cleared by hardware. Note: When the value of this bit is "0", the hardware automatically shields the software write SBUF operation.
		1: TE interrupt is valid 0: TE interrupt is invalid
2	FE	Receive frame error flag bit, set by hardware, cleared by software 1: FE interrupt is valid 0: FE interrupt is invalid
1	TI	Send completion interrupt flag bit, set by hardware, cleared by software 1: TI interrupt is valid 0: TI interrupt is invalid
0	RI	Receive complete interrupt flag bit, set by hardware, cleared by software 1: RI interrupt is valid 0: RI interrupt is invalid

Page 400

16.9.6 Interrupt flag bit clear register (LPUART_ICR)

Offset address: 0x14

Reset value: 0x0000 0007



Bit symbol describe

31:3	Reserved	
2	FECLR	Clear the received frame error flag; Write 0 to clear zero, write 1 invalid
1	TICLR	Clear the sending complete interrupt flag bit; Write 0 to clear zero, write 1 invalid
0	RICLR	Clear the reception completion interrupt flag bit; write 0 to clear it, write 1 to have no effect

17 I2C bus (I2C)

17.1 Introduction

I2C is a two-wire bidirectional synchronous serial bus. It uses a clock line and a data line to connect two devices on the bus.

The transfer of information between devices provides a simple and efficient method for data exchange between devices. Each connected to

The devices on the bus have unique addresses. Any device can be used as either a master or a slave, but at the same time

Only one host is allowed. The I2C standard is a true multi-master with a conflict detection mechanism and an arbitration mechanism.

Machine bus, it can use the arbitration mechanism to avoid data conflicts and protect data when multiple hosts request control of the bus at the same time

The I2C bus controller can meet the various specifications of the I2C bus and support all the transmission modes of communication with the I2C bus.

I2C logic can handle byte transfers autonomously. It can keep track of serial transfers, and there is also a status register

(I2C_STAT) can reflect the status of I2C bus controller and I2C bus.

17.2 Main features

The I2C controller supports the following features:

Support four working modes of master sending/receiving and slave sending/receiving

- Support standard (100Kbps) / fast (400Kbps) / high speed (1Mbps) three working rates
- Support 7-bit addressing function
- Support noise filtering function
- Support broadcast address
- Support interrupt status query function

17.3 Protocol description

The I2C bus uses the "SCL" (Serial Clock Bus) and "SDA" (Serial Data Bus) connecting devices to transmit information. host
The machine outputs the serial clock signal on the SCL line, and the data is transmitted on the SDA line. Each byte (up to
Bit MSB start transmission), followed by an acknowledge bit. One SCL clock pulse transfers one data bit.

17.3.1 Data Transmission on the I2C Bus

Usually the standard I2C transmission protocol consists of four parts: start (S) or repeated start signal (Sr), slave address and read and write
Bit, transfer data, stop signal (P).

SDA



Figure 17-1 I2C transmission protocol

Start signal, repeat start signal, stop signal

When the bus is in an idle state (SCL and SDA lines are high at the same time), a signal from high to low appears on the SDA line.

Number, indicating that a start signal has been generated on the bus.

When there is no stop signal between the two start signals, a repeated start signal is generated. The host uses this method

Communicate with another slave or the same slave in different transmission directions (for example: from writing device to slave device

Read) without releasing the bus.

When the SCL line is high, a low-to-high signal appears on the SDA line, which is defined as a stop signal. Host to total

The line sends a stop signal to end the data transmission.



SDA

Figure 17-2 START and STOP conditions

Slave address and read/write bit

When the start signal is generated, the master immediately transmits the first byte of data: 7-bit slave address + read and write bits, read and write

The bit controls the data transmission direction of the slave (0: write; 1: read). The slave opportunity addressed by the master passes in the 9th

Page 403

The SCL clock cycle sets SDA low as a response.

Data transmission

During data transmission, one SCL clock pulse transmits one data bit, and the SDA line only works when SCL is low.

It can be changed only when.

SDA

SCL



Figure 17-3 I2C bus upper transfer

17.3.2 Acknowledge on the I2C bus

Each transmission of a byte is followed by an acknowledge bit. By pulling the SDA line low to allow the receiving end to respond to the transmission end. ACK is a low-level signal. When the clock signal is high, SDA remains low, indicating that the receiving end has completed. The function receives the data from the sender.

When the host is used as a transmitting device, if a non-response signal (NACK) is generated on the slave, the host can generate a stop signal To exit the data transmission, or generate a repeated start signal to start a new round of data transmission. When the host acts as a receiver

When a non-response signal (NACK) occurs when the device is in the process, the slave releases the SDA line, causing the master to generate a stop signal.

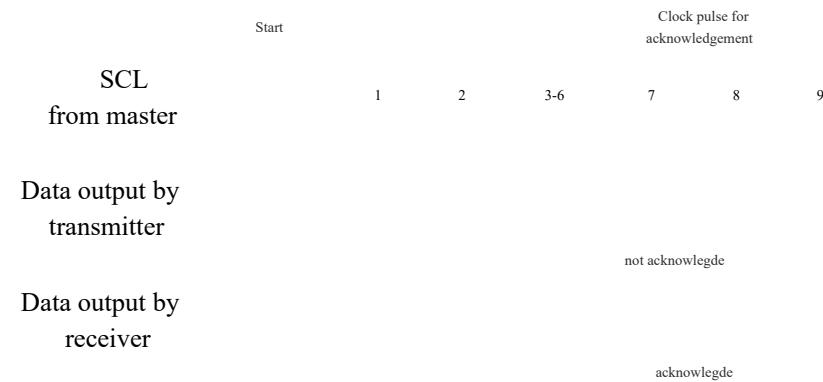
Page 404

Figure 17-4 Acknowledge signal on the I2C bus

17.3.3 Arbitration on the I2C Bus

Arbitration on the I2C bus is divided into two parts: synchronization on the SCL line and arbitration on the SDA line

Synchronization on SCL line (clock synchronization)

Since the I2C bus has the logic function of line "and", as long as one node on the SCL line sends a low level, the total

The line is low level. When all nodes are sending high level, the bus can behave as high level. so,

The low level time of the clock is determined by the device with the longest clock level period, and the high level time of the clock is determined by

The device with the shortest normal period is determined. Due to the characteristic of I2C, when multiple hosts send clock signals at the same time,

Shown on the line is a unified clock signal. If the slave wants the master to reduce the transmission speed, it can pass the SCL master

Pull down to extend its low level time to notify the host, when the host finds the level of SCL when preparing for the next transmission

When pulled low, wait until the slave completes the operation and releases control of the SCL line.

SDA online arbitration

The arbitration on the SDA line is also due to the logic function of the line "AND" on the I2C bus. After the host sends data,

Determine whether to withdraw from the competition by comparing the data on the bus. The host that loses arbitration immediately switches to una

To ensure that it can be addressed by the host that wins the arbitration. When the host that has failed arbitration continues to output

Clock pulse (on SCL) until the current serial byte is sent. Through this principle, I2C total

The line ensures that data is not lost when multiple hosts attempt to control the bus.

Page 405

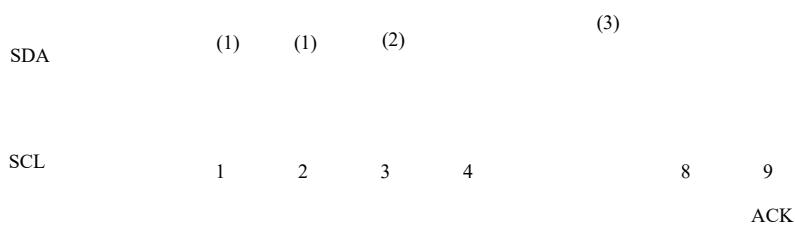


Figure 17-5 Arbitration on the I2C bus

- a) Another device sends serial data;
- b) Another device first cancels a logic 1 (dotted line) sent by the I2C host by pulling SDA low. arbitration
Lost, I2C enters the slave receiving mode;
- c) At this time, I2C is in slave receiving mode, but it still generates clock pulses until the current byte is sent. I2C will
No clock pulse is generated for the transfer of the next byte. Once the arbitration is won, the data transmission on SDA is changed by the new
The host starts.

17.4 Functional description

The I2C bus uses two wires between the devices connected to the bus "SCL" (Serial Clock Line) and "SDA" (Serial Data Line)

Send information. Filtering logic can filter glitches on the data bus to protect the integrity of the data. Because there is only no direction

For ports, I2C components need to use open-drain buffers to the pins. Every device connected to the bus can use software

The file is addressed by a specific address. The I2C standard is a real multiplicity with a conflict detection mechanism and an arbitration mechanism. Host bus. It can prevent data conflicts when two or more hosts start to transmit data at the same time. I2C bus
The status can be queried in the status register.

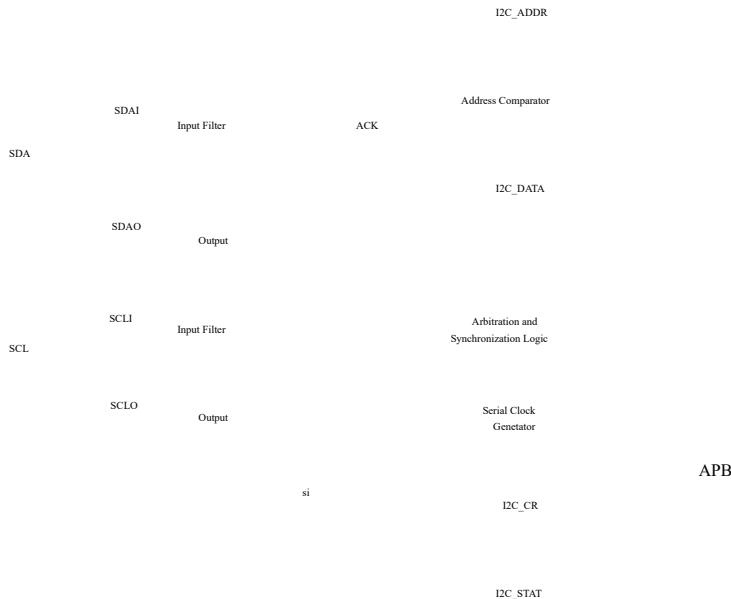


Figure 17-6 I2C function module diagram

17.4.1 Serial Clock Generator

The serial clock generator uses an 8-bit counter as the baud rate generator, SCL signal and PCLK signal

The frequency relationship of $F_{SCL} = F_{PCLK} / 8 / (I2C_TM.tm + 1)$, where $I2C_TM.tm$ should be greater than 0.

The following table lists the output frequency value of the SCL signal when the PCLK frequency is combined with $I2C_TM.tm$.

PCLK (KHz)	I2C_TM.tm						
	1	2	3	4	5	6	7
1000	62	41	31	25	20	17	15
2000	125	83	62	50	41	35	31
4000	250	166	125	100	83	71	62

User manual							
6000	375	250	187	150	125	107	93
8000	500	333	250	200	166	142	125
10000	625	416	312	250	208	178	156
12000	750	500	375	300	250	214	187
14000	875	583	437	350	291	250	218
16000	1000	666	500	400	333	285	250

Table 17-1 I2C clock signal baud rate

17.4.2 Input Filter

The input signal is synchronized with PCLK, and the spike pulse signal lower than the period of PCLK will be filtered out.

When this module is used as the host, if the value of I2C_TM is less than or equal to 9, I2C_CR.H1M should be set to 1;

If the value of I2C_TM is greater than 9, I2C_CR.H1M should be set to 0.

When this module is used as a slave, if the ratio of PCLK to SCL frequency is less than or equal to 30, I2C_CR.H1M should be set

If the ratio of PCLK to SCL frequency is greater than 30, I2C_CR.H1M should be set to 0.

17.4.3 Address Comparator

The I2C comparator compares its own slave address with the received 7-bit slave address. It can use "I2C_ADDR"

The register programs its own slave address. And according to the "i2cadr" bit of the "I2C_ADDR" register and the first

Compare the received 8-bit byte or broadcast address (0x00). If any of them are the same, "I2C_CR" is registered

The "si" bit of the device will be set to 1 and an interrupt request will be generated.

17.4.4 Response flag

The "aa" flag bit of the "I2C_CR" register is the response flag bit. When the "aa" bit is 1, the I2C module will respond after receiving the data

Acknowledge bit, when the "aa" bit is 0, the I2C module will return the non-acknowledge bit after receiving the data.

17.4.5 Interrupt Generator

The "si" flag bit in the "I2C_CR" register is an interrupt flag bit. Whenever the value of the status register (I2C_STAT) is generated

When changing (except changing to 0xF8), the "si" flag will be set to 1. When an interrupt is generated, by querying the status register

The device (I2C_STAT) can learn the status of the I2C bus to determine the actual source of the interrupt. In order to proceed to the next step

To operate, the "si" flag must be cleared by software.

17.4.6 Working Mode

The I2C component can realize 8-bit bidirectional data transmission, and the transmission rate can reach 100Kbps in the standard mode, and the transm

Up to 400Kbps in high-speed mode, up to 1Mbps in ultra-high-speed mode, and can work in four modes:

Host sending mode, host receiving mode, slave receiving mode, and slave sending mode. There is also a special mode

The call mode is similar to that of the slave receiving mode.

Host sending mode

The master sends multiple bytes to the slave, and the master generates the clock, so it is necessary to fill in the set value in I2C_TM. host

I2C_CR.sta needs to be set to 1 in the machine transmission mode. When the bus is free, the host initiates a start bit START.

If successful, I2C_CR.si is set to 1. Next, write the slave address and write bit (SLA+W) into I2C_DATA,

After clearing the "si" bit, SLA+W is issued on the bus.

After the master sends out SLA+W and receives the slave acknowledge bit ACK, "si" is set to 1. Next according to the user-defined format send data. After all the data is sent, set I2C_CR.sto to 1, clear the "si" bit and send a STOP signal,

It is also possible to send a repeated start signal for a new round of data transmission.

Page 409

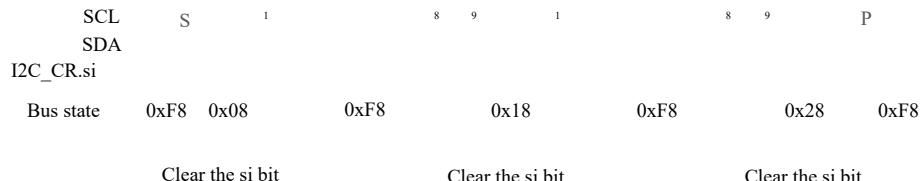
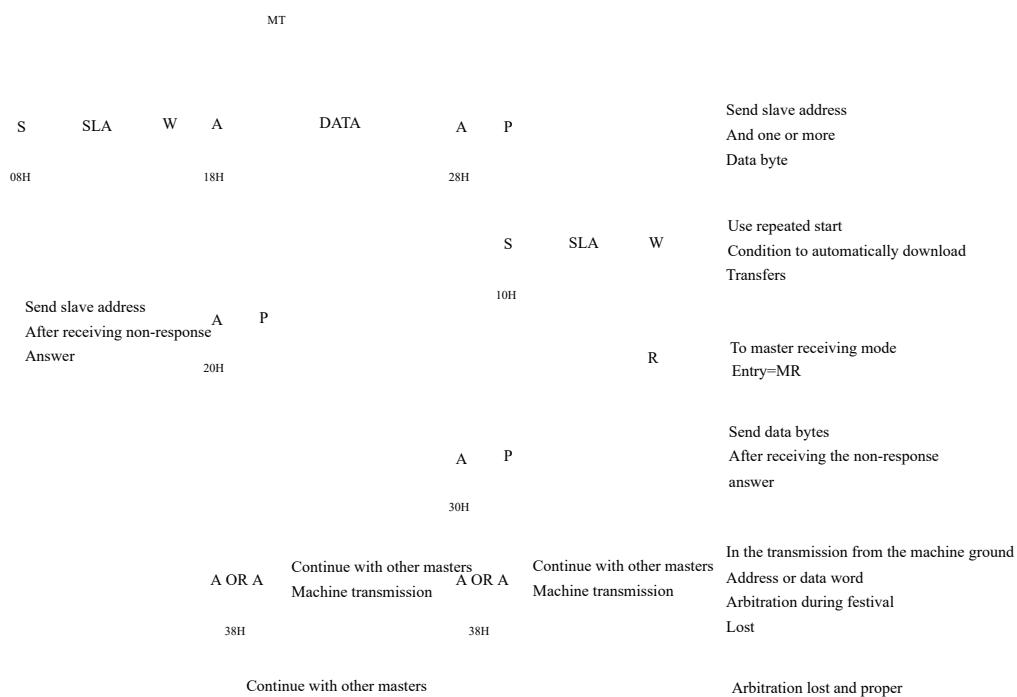


Figure 17-7 Data synchronization diagram in master sending mode



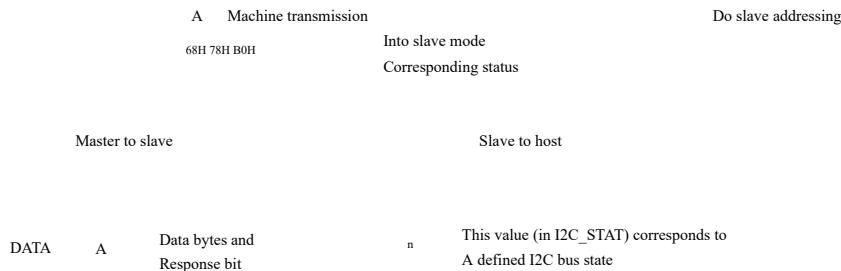


Figure 17-8 I2C host sending state diagram

Page 410**Host receiving mode**

In the receiving mode of the master, the data is transmitted by the slave. The initial setting is the same as the host sending state. After the bit, I2C_DATA should be written into the slave address and “read bit” (SLA+R). Receive slave acknowledge bit ACK. Then I2C_CR.si is set to 1. After "si" is cleared to 0, it starts to receive data from the slave. If I2C_CR.aa is 1, the master receives The response bit is returned after the data; if it is 0, the host does not respond with NACK after receiving the data. Then the host can send a stop letter or repeat the start signal to start the next round of data transmission.

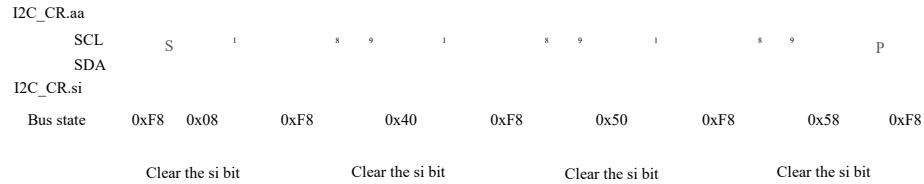


Figure 17-9 Data synchronization diagram in master receiving mode

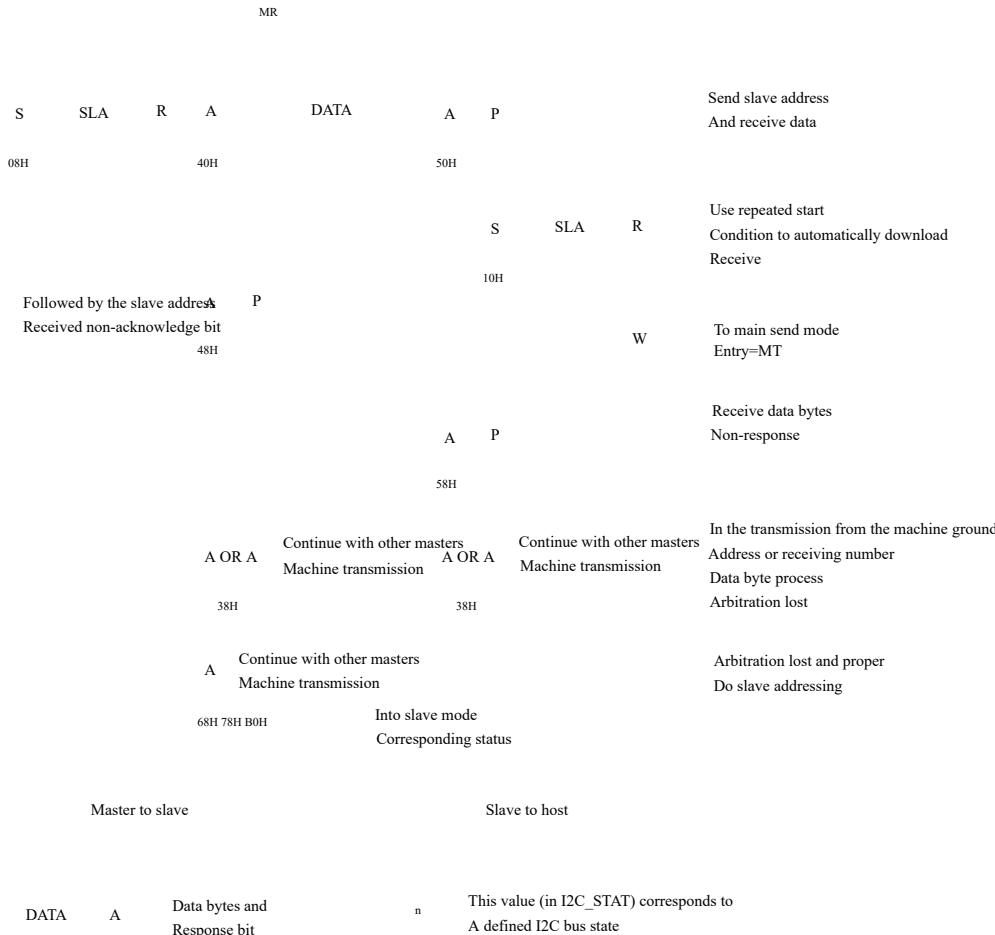
Page 411

Figure 17-10 I2C host receiving state diagram

Slave receiving mode

In the slave receiving mode, the slave receives the data from the host. Before the start of the transfer, I2C_ADDR should be written

Slave address, I2C_CR.aa is set to 1 to respond to the addressing of the master. After the above initialization, the slave enters the idle mode

Type, waiting for the "write" signal (SLA+W). If the master fails in arbitration, it will directly enter the slave receiving mode.

When the slave is addressed by the "write" signal SLA+W, the "si" bit needs to be cleared to receive data from the master. like

If I2C_CR.aa=0 during transmission, the slave will return NACK in the next byte, and the slave will also

Turn to an unaddressed slave, terminate the contact with the master, no longer receive data, and I2C_DATA keeps the previous received

The data. The slave address recognition can be restored by setting the "aa" bit, which means that the "aa" bit can temporarily remove the I2C module
Separated on the I2C bus.

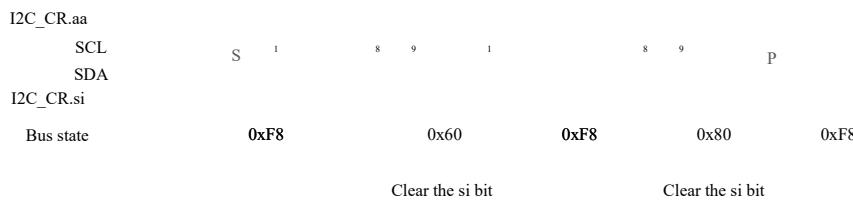


Figure 17-11 Data synchronization diagram in slave receiving mode

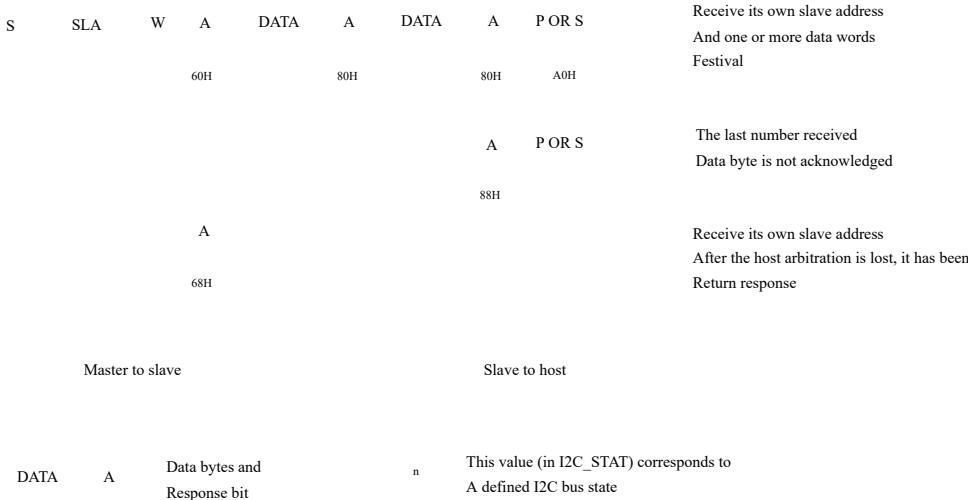


Figure 17-12 Slave receiving state diagram

Slave sending mode

In the slave sending mode, the data is sent from the slave to the master. After initializing the I2C_ADDR and I2C_CR.aa values,

The device waits until its own address is addressed by the "read" signal (SLA+R). If the master fails in arbitration, you can also enter the slave

Machine sending mode.

When the slave is addressed by the "read" signal SLA+R, "si" needs to be cleared to send data to the master. Usually host
An acknowledge bit will be returned after receiving each byte of data.
If I2C_CR_aa is cleared during the transmission, the slave will send the last byte of data, and in the next transmission
Send all 1 data during the input and turn itself into an unaddressed slave.

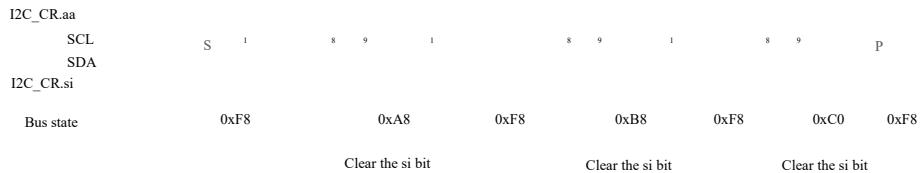


Figure 17-13 Data synchronization diagram in slave sending mode

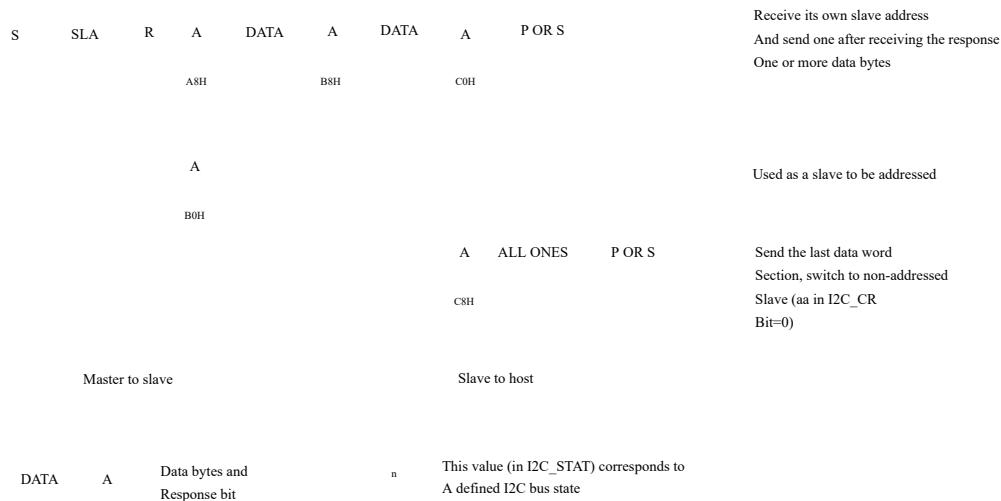
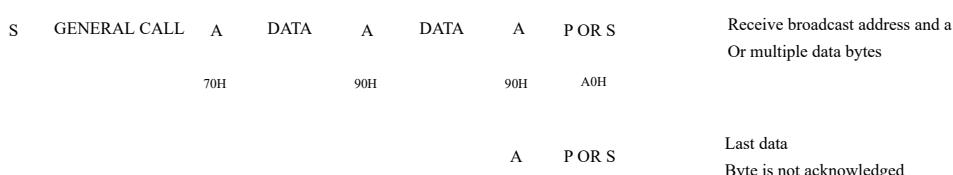


Figure 17-14 I₂C slave sending state diagram

Page 414

Broadcast call mode

The general call mode is a special slave receiving mode, the addressing mode is 0x00, the slave address and reading and writing are both 0. When both I2C_ADDR.GC and I2C_CR.aa are set to 1, the general call mode is enabled. In this mode The value of I2C_STAT is different from the value of I2C_STAT in normal slave receiving mode. Arbitration failure may also enter the broadcast Call mode.



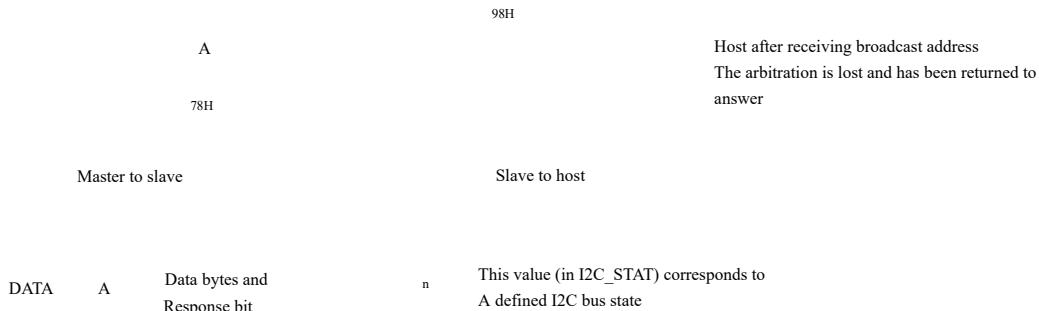


Figure 17-15 I2C broadcast call status diagram

Page 415**17.4.7 State code expression**

There are two special states in the I2C status register: F8H and 00H.

F8H: This status code means that there is no relevant information available because the serial interrupt flag "si" has not been set.

This situation occurs between other states and the I2C module has not yet started to perform serial transmission.

00H: This status code indicates that a bus error occurred during I2C serial transmission. When the illegal position of the format frame

A bus error occurs when a start or stop condition occurs on the device. These illegal locations refer to the locations in the serial transmission process.

Address byte, data byte or response bit. When external interference affects the internal I2C module signal, a bus error will also occur.

error. "Si" is set when a bus error occurs.

Status code	describe
Main send mode	
08H	Start condition sent
10H	Repeated start condition sent
18H	SLA+W has been sent, ACK has been received
20H	SLA+W has been sent, non-ACK has been received
28H	The data in I2C_DATA has been sent, and ACK has been received

30H	The data in I2C_DATA has been sent, but not ACK has been received
38H	Lost arbitration when SLA+ reads, writes or writes data bytes

Master receiving mode

08H	Start condition sent
10H	Repeated start condition sent
38H	Arbitration lost in non-ACK
40H	SLA+R has been sent, ACK has been received
48H	SLA+R has been sent, non-ACK has been received
50H	Data byte has been received, ACK has been returned
58H	Data byte has been received, non-ACK has been returned

Slave receiving mode

60H	Received own SLA+W and returned ACK
-----	-------------------------------------

Page 416

68H	When the master is in the SLA+read and write lost arbitration, it has received its own SLA+W, and has returned ACK
80H	The previous addressing used its own slave address, data bytes have been received, and ACK has been returned
88H	The previous addressing used its own slave address, data bytes have been received, and non-ACK has been returned
A0H	When static addressing, stop condition or repeated start condition is received

Slave send mode

A8H	Received own SLA+R and returned ACK
B0H	When the host loses arbitration, has received its own SLA+R, and has returned ACK
B8H	Data has been sent, ACK has been received
C0H	Data byte has been sent, non-ACK has been received
C8H	The loaded data byte has been sent and ACK has been received

Broadcast call mode

70H	Broadcast address (0x00) has been received, ACK has been returned
78H	The master control loses arbitration in SLA+ reading and writing, has received the broadcast address, and has returned ACK
90H	The broadcast address was used for the previous addressing, data bytes have been received, and ACK has been returned
98H	The broadcast address was used for the previous addressing, data bytes have been received, and non-ACK has been returned
A0H	When static addressing, stop condition or repeated start condition is received

Other miscellaneous status

F8H	No relevant status information available, si=0
00H	Bus error occurred during transmission, or external interference caused I2C to enter an undefined state

Table 17-2 I2C status code description

Page 417

17.5 Programming Example

17.5.1 Host sending example

Step1: According to the relevant description of the pin digital multiplexing function in the GPIO chapter, map SCL and SDA to the required pins;

And configure the SCL and SDA pins as open-drain output mode.

Step2: Set PERI_CLKEN.I2C to 1, enable the I2C module clock.

Step3: Write 0 and 1 to PERI_RESET.I2C in turn to reset the I2C module.

Step4: Configure I2C_TM so that the clock rate of SCL meets the application requirements.

Step5: Set I2C_TMRUN to 1, and enable the SCL clock generator.

Step6: Set I2C_CR.ens to 1, enable the I2C module.

Step7: Set I2C_CR.sta to 1, the bus tries to send a Start signal.

Step8: Wait for I2C_CR.si to become 1, and the Start signal has been sent to the bus.

Step9: Query I2C_STAT, if the register value is 0x08 or 0x10, continue to execute the next step, otherwise proceed

Line error handling.

Step10: Write SLA+W to I2C_DATA, set I2C_CR.sta to 0, set I2C_CR.si to 0,

Send SLA+W.

Step11: Wait for I2C_CR.si to change to 1, and SLA+W has been sent to the bus.

Step12: Query I2C_STAT, if the register value is 0x18, continue to the next step. Otherwise go wrong
deal with.

Step13: Write the data to be sent to I2C_DATA, set I2C_CR.si to 0, and send the data.

Step14: Wait for I2C_CR.si to change to 1, and the data has been sent to the bus.

Step15: Query I2C_STAT, if the register value is 0x28, continue to the next step. Otherwise go wrong
deal with.

Step16: If the data to be sent is not completed, jump to Step13 to continue execution.

Step17: Set I2C_CR.sto to 1, set I2C_CR.si to 0, and the bus tries to send a Stop signal.

Step18: Wait for I2C_CR.si to change to 1, Stop signal has been sent to the bus.

Page 418**17.5.2 Host reception example**

Step1: According to the relevant description of the pin digital multiplexing function in the GPIO chapter, map SCL and SDA to the required pins; And configure the SCL and SDA pins as open-drain output mode.

Step2: Set PERI_CLKEN.I2C to 1, enable the I2C module clock.

Step3: Write 0 and 1 to PERI_RESET.I2C in turn to reset the I2C module.

Step4: Configure I2C_TM so that the clock rate of SCL meets the application requirements.

Step5: Set I2C_TMRUN to 1, and enable the SCL clock generator.

Step6: Set I2C_CR.ens to 1, enable the I2C module.

Step7: Set I2C_CR.sta to 1, the bus tries to send a Start signal.

Step8: Wait for I2C_CR.si to become 1, and the Start signal has been sent to the bus.

Step9: Query I2C_STAT, if the register value is 0x08 or 0x10, continue to the next step, otherwise proceed Error handling.

Step10: Write SLA+R to I2C_DATA, set I2C_CR.sta to 0, set I2C_CR.si to 0, and send Send SLA+R.

Step11: Wait for I2C_CR.si to change to 1, and SLA+R has been sent to the bus.

Step12: Query I2C_STAT, if the register value is 0x40, continue to the next step, otherwise proceed to error handling reason.

Step13: Set I2C_CR.aa to 1, enable the response flag.

Step14: Set I2C_CR.si to 0, the slave sends data, and the master sends ACK or NACK according to I2C_CR.aa.

Step15: Wait for I2C_CR.si to change to 1, and read the received data from I2C_DATA.

Step16: Query I2C_STAT, if the register value is 0x50 or 0x58, continue to the next step, no Then carry out error handling.

Step17: If the data to be received is only the last byte, set I2C_CR.aa to 0 and enable the non-response flag.

Step18: If the data to be received is not completed, jump to Step14 to continue execution.

Step19: Set I2C_CR.sto to 1, set I2C_CR.si to 0, and the bus tries to send a Stop signal.

Step20: Wait for I2C_CR.si to change to 1, Stop signal has been sent to the bus.

17.5.3 Slave receiving example

Step1: According to the relevant description of the pin digital multiplexing function in the GPIO chapter, map SCL and SDA to the required pins;
And configure the SCL and SDA pins as open-drain output mode.

Step2: Set PERI_CLKEN.I2C to 1, enable the I2C module clock.

Step3: Write 0 and 1 to PERI_RESET.I2C in turn to reset the I2C module.

Step4: Set I2C_CR.ens to 1, enable the I2C module.

Step5: Configure I2C_ADDR as the slave address.

Step6: Set I2C_CR.aa to 1, enable the response flag.

Step7: Wait for I2C_CR.si to change to 1 and be addressed by SLA+W.

Step8: Query I2C_STAT, if the register value is 0x60, continue to the next step, otherwise an error occurs
deal with.

Step9: Set I2C_CR.si to 0, the master sends data, and the slave returns ACK or NACK according to I2C_CR.aa.

Step10: Wait for I2C_CR.si to change to 1, and read the received data from I2C_DATA.

Step11: Query I2C_STAT, if the register value is 0x80, continue to the next step, otherwise an error occurs
deal with

Step12: If the data to be received is not completed, skip to Step9 to continue execution.

Step13: Set I2C_CR.aa to 0, and set I2C_CR.si to 0.

17.5.4 Slave sending example

Step1: According to the relevant description of the pin digital multiplexing function in the GPIO chapter, map SCL and SDA to the required pins;
And configure the SCL and SDA pins as open-drain output mode.

- Step2: Set PERI_CLKEN.I2C to 1, enable the I2C module clock.
- Step3: Write 0 and 1 to PERI_RESET.I2C in turn to reset the I2C module.
- Step4: Set I2C_CR.ens to 1, enable the I2C module.
- Step5: Configure I2C_ADDR as the slave address.
- Step6: Set I2C_CR.aa to 1, enable the response flag.
- Step7: Wait for I2C_CR.si to change to 1, and be addressed by SLA+R.
- Step8: Query I2C_STAT, if the value of this register is 0xA8, continue to the next step, otherwise proceed to output Wrong processing.
- Step9: Write the data to be sent to I2C_DATA, set I2C_CR.si to 0, and send the data.
- Step10: Wait for I2C_CR.si to change to 1, and the data has been sent to the bus.
- Step11: Query I2C_STAT, if the value of this register is 0xB8 or 0xC0, continue to the next step, Otherwise, perform error handling.
- Step12: If the data to be sent is not completed, jump to Step9 to continue execution.
- Step13: Set I2C_CR.aa to 0, and set I2C_CR.si to 0.

17.6 Register description

Register list

I2C base address: 0x40000400

Offset	Register name	access	Register description
0x00	I2C_TMRUN	RW	I2C baud rate counter enable register.
0x04	I2C_TM	RW	I2C baud rate counter configuration register.
0x08	I2C_CR	RW	I2C configuration register.
0x0c	I2C_DATA	RW	I2C data register.
0x10	I2C_ADDR	RW	I2C address register.
0x14	I2C_STAT	RO	I2C status register.

Table 17-3 Register List

17.6.1 I2C baud rate counter enable register (I2C_TMRUN)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twent	four	thirt	twent	twent	o20	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	tme
Reserved																
																RW

Bit	mark	Function description
-----	------	----------------------

31:1 Reserved

0	tme	Baud rate counter enable. 0-disable	1-Enable
---	-----	-------------------------------------	----------

17.6.2 I2C baud rate counter configuration register (I2C_TM)

Address offset: 0x04

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twent	four	thirt	twent	twent	o20	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	tm
Reserved																
																RW

Bit	mark	Function description
-----	------	----------------------

31:8 Reserved

7:0	tm	tm: Baud rate counter configuration value.
-----	----	--

$F_{SCL} = F_{PCLK} / 8 / (tm + 1)$, where $tm > 0$

Page 423**17.6.3 I2C Configuration Register (I2C_CR)**

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	twenty four	enty thir	enty twen	ty o 10	19	18	17	16
Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
														0
Reserved														
														RW

Bit mark Function description

31:7 Reserved

6 ens I2C module enable control

0-prohibited

1-Enable

5 sta I2C bus control

0-No function

1-Send a START to the bus

4 sto I2C bus control

0-No function

1-Send a STOP to the bus

3 si I2C interrupt flag

Read 1, I2C interrupt has occurred

Write 0, I2C performs one step operation

2 aa Response control bit

0-Send NAK

1-Send ACK

1 Reserved

0 h1m I2C filter parameter configuration

Page 424

0-Advanced filtering, higher anti-interference performance

1-Simple filtering, faster communication rate

Note: For details, please refer to the [Input Filter] chapter.

Page 425**17.6.4 I2C Data Register (I2C_DATA)**

Address offset: 0x0c

Reset value: 0x00000000

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16
----	----	----	----	----	----	----	-------------	--------------	------------	------------	----	----	----	----

Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
----	----	----	----	----	----	---	---	---	---	---	---	---	---	---	---

i2cdat

Reserved

RW

Bit	mark	Function description
-----	------	----------------------

31:8 Reserved

7:0 i2cdat I2C data register

In I2C sending mode, write the data to be sent

In I2C receiving mode, read the received data

17.6.5 I2C Address Register (I2C_ADDR)

Address offset: 0x10

Reset value: 0x00000000

31	30	29	28	27	26	25	twenty four	twenty five	twenty six	twenty seven	twenty eight	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i2cadr															
Reserved															
												RW			RW

Bit	mark	Function description
-----	------	----------------------

31:8 Reserved

7:1 i2cadr I2C slave mode address.

0 GC Broadcast address response enable

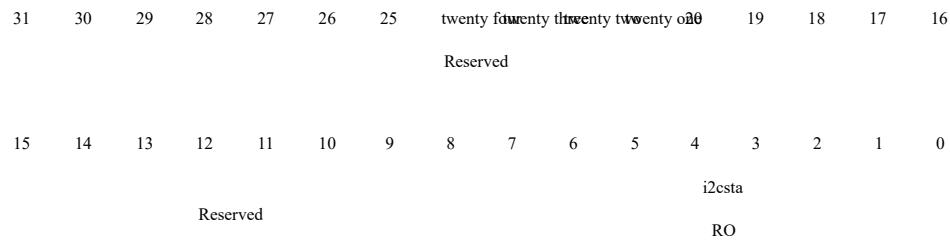
0-prohibited

1-Enable

17.6.6 I2C Status Register (I2C_STAT)

Address offset: 0x14

Reset value: 0x00000000



18 Serial Peripheral Interface (SPI)

18.1 Introduction to SPI

The SPI interface is a synchronous serial data communication interface working in full-duplex mode, using 4 pins for communication: MISO, MOSI, SCK, CS/SSN. When the SPI is used as the master, the CS and SCK signals are output to control the communication. Letter process. When SPI is used as a slave, it communicates under the control of SSN and SCK signals.

18.2 SPI main features

Support SPI master mode, SPI slave mode

Support standard four-wire full-duplex communication

Support configuration of serial clock polarity and phase

Host mode supports 7 communication rates

The maximum frequency division factor of the host mode is PCLK/2, and the maximum communication rate is 16M bps

The maximum frequency division factor of slave mode is PCLK/4, and the maximum communication rate is 12M bps

The frame length is fixed at 8 bits, and the MSB is transmitted first

18.3 SPI function description

18.3.1 SPI Master Mode

The length of each data frame is fixed at 8 bits, and the first bit of the transmitted data is fixed at MSB. Set SPI_CR.mstr

If it is 1, the SPI interface works in master mode. Write data into the SPI_Data register to start SPI transmission.

The SCK pin will automatically generate the serial clock; on the edge of the serial clock, the data in the shift register is sent to

The data of MOSI pin and MISO pin are received into the shift register. Each time the transmission of a data frame is completed,

SPIF will be set by hardware. Reading SPI_DATA can clear the SPIF flag. The frequency of the SCK pin output clock is determined by

SPI_CR[spr2:spr0] controls, the output frequency range is PCLK /2~PCLK/128; the output of CS pin

The output level is controlled by SPI_SS.ssn, and the output level of the GPIO pin is controlled by the GPIO related register.

The typical application block diagram of the host mode is shown below.



MISO	
MOSI	Slave 1
SCK	
NCS	

The communication in the host mode is shown in the figure below, where CPOL=0 and CPHA=0.



18.3.2 SPI Slave Mode

The length of each data frame is fixed at 8 bits, and the first bit of received data is fixed at MSB. Set SPI_CR.mstr

If it is 0, the SPI interface works in slave mode. In this mode, the SCK pin is used as an input pin and the serial clock comes from

For the external host; the SSN pin is used as an input pin and the chip select signal comes from the external host or is fixed at low level. SSN cited

See the auxiliary registers in the GPIO chapter for details of the pins. The typical application block diagram of the slave mode is shown below.



When the SPI slave receives a data frame from the SPI master, the SPIF bit will be set high; the user program should read it as soon as possible

Data received. The communication sequence of receiving data in slave mode is shown below, where CPOL=0 and CPHA=0.



Figure 18-1 Schematic diagram of slave receiving

When the SPI slave needs to send data to the master, the master should send data to the SPI_DATA register as soon as possible after the master pulls dc

Write the first byte of data to be sent in the file; whenever the SPIF flag is 1, you should read SPI_DATA as soon as possible

To clear the SPIF flag, and write subsequent data to be sent to the SPI_DATA register. Number of sending in slave mode

The data communication sequence is shown below, where CPOL=0 and CPHA=0.



Figure 18-2 Schematic diagram of slave sending

18.3.3 SPI Data Frame Format

The SPI interface frame format depends on the configuration of the clock polarity bit CPOL and the clock phase bit CPHA.

When CPOL is 0, the idle state of the SCK line is low. When CPOL is 1, the idle state of SCK line is high

Level. When CPHA is 0, the data will be sampled when the first SCK clock transition signal transitions. When CPHA

When it is 1, the data will be sampled when the second SCK clock signal transitions.

The SPI interface host frame format is shown in the figure below.

Figure 18-3 Host mode frame format

The SPI interface slave frame format is shown in the figure below.

Figure 18-4 Data frame format when slave CPHA is 0

Page 432

Figure 18-5 Data frame format when the slave CHPA is 1

18.3.4 SPI Status Flags and Interrupts

The SPI will generate the following three status flags during operation. The conditions and clearing methods are as follows.

When the transmission of a data frame is completed, SPIF will be set by hardware. Read the SPI_DATA register

To clear the flag bit.

When SPI works in host mode and the external SSN input is low, SPI_STAT.mdf will be set by hardware,

It means that there are other SPI masters occupying the bus. When the SSN input is high, SPI_STAT.mdf will be hardened
Files are automatically cleared.

When SPI works in slave mode, if the SSN pin is pulled high during data transmission, then

SPI_STAT.sserr will be set. Set SPI_CR.spen to 0 to clear this flag.

If the SPI interrupt vector is enabled, an interrupt can be generated in the following two situations:

SPI transmission is complete, that is, SPI_STAT.SPIF is 1

The host mode of SPI is wrong, that is, SPI_STAT.mdf is 1

18.3.5 SPI multi-machine system configuration instructions

When the SPI module acts as a host and works in a single-host system, it can be controlled by the SPI_CS pin or GPIO pin
System from the machine. When selecting the SPI_CS pin as the chip select signal of the slave, set SPI_SSNS.ssn to 0.

Select the slave and set SPI_SSNS.ssn to 1 to release the slave. When selecting the GPIO pin as the slave chip

When selecting a signal, set the corresponding bit of the GPIOx_OUT register to 0 to select the slave and set GPIOx_OUT
The corresponding bit of the register is 1 to release the slave.

When the SPI module is a slave, configure the source of SPI_SSNS as required (see GPIO port auxiliary control for details)
器). When SSN is low, the slave can be selected for communication; when SSN is high, the slave is in the off state.

Selected state.

When working in SPI mode with multiple masters and multiple slaves, all slave chip select signals are connected through GPIO pins.

The host must also connect to the SSN signal of other hosts through the GPIO pin to monitor whether the bus is occupied.

The operation method that Master0 needs to communicate as shown in the figure below is: wait for Master0.SSN to become high; from GPIO0 outputs low to notify Master1 to release the SPI bus; outputs low from GPIO2 to select Slave1; and Slave1 communicates; output high from GPIO2 to release Slave1; output high from GPIO0 to release Master1.

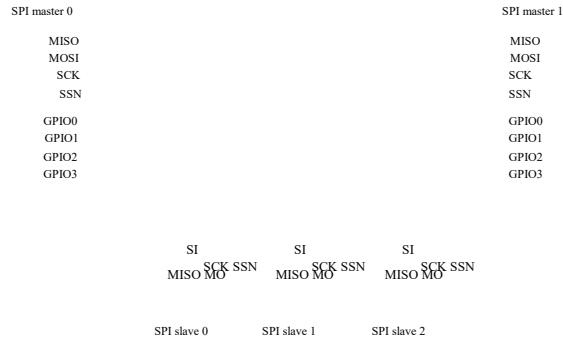


Figure 18-6 Schematic diagram of SPI multi-master/multi-slave system

18.3.6 SPI pin configuration description

SPI can maintain some or all of its functions under some special pin configurations.

The specific situation is as follows ("√" represents that the pin is configured and used, and blank represents that the pin is not configured):

SPI_CS (Master)/ SPI_SS (slave)	SCK	MOSI	MISO	Function Description
Host mode	√	√	√	General configuration
		√	√	All hosts are functioning normally
		√		All hosts are functioning normally
	√	√		Host sending function is normal
	√	√	√	Host receiving function is normal
		√		Host sending function is normal
			√	Host receiving function is normal
	√	√	√	General configuration
Slave mode		√	√	All slaves function normally
	√	√	√	Slave receiving function is normal
	√	√		Slave sending function is normal
	√			All slaves function normally
	Fixed low level	√	√	Slave receiving function is normal
	√	√		Slave sending function is normal
	Fixed low level	√		
	√		√	

Table 18-1 SPI pin configuration description table

Notice:

- Situations not listed in the table are not currently supported.
- In the host mode, even if you do not use the SPI_CS chip select output, you need to set SPI.SSN to 1 before sending data.
 Need to set SPI.SSN to 0 after sending data.
- In slave mode and chip select input is fixed at low level, in order to maintain normal functions, SPI_CR.cpha=1 must be satisfied.

18.4 SPI programming example

18.4.1 SPI master sending example

Step1: According to the relevant description of the pin digital multiplexing function in the GPIO chapter, map CS/SCK/MISO/MOSI to Needed pins; and configure the CS/SCK/MOSI pin as output mode, and configure the MISO pin as input mode.

Step2: Set SPI_CR.mstr to 1, make SPI work in master mode.

Step3: Configure SPI_CR[spr2:spr0] to make the clock rate of SCK output meet the application requirements.

Step4: Configure SPI_CR.cpol and SPI_CR.cpha to make the data frame format meet the application requirements.

- Step5: Set SPI_CR.spen to 1, and enable the SPI interface.
- Step6: Set SPI_SSN.ssn to 0, make the CS pin output low level to select the slave.
- Step7: Write the data to be sent into SPI_DATA, and wait for SPIF to become 1.
- Step8: Read SPI_DATA to clear the SPIF flag.
- Step9: If the data to be sent is not completed, jump to Step7 to continue execution.
- Step10: Set SPI_SSN.ssn to 1, make the CS pin output high level to release the slave.

Notice:

- GPIO can be used instead of CS to realize chip select output, which is mostly used in multi-computer communication systems.
- SPI_SSN.ssn must be set to 0 during the transfer process, and SPI_SSN.ssn must be set to 1 after the transfer is completed.

18.4.2 SPI master receiving example

- Step1: According to the relevant description of the pin digital multiplexing function in the GPIO chapter, map CS/SCK/MISO/MOSI to Needed pins; and configure the CS/SCK/MOSI pin as output mode, and configure the MISO pin as input mode.
- Step2: Set SPI_CR.mstr to 1, make SPI work in master mode.
- Step3: Configure SPI_CR[spr2:spr0] to make the clock rate of SCK output meet the application requirements.
- Step4: Configure SPI_CR.cpol and SPI_CR.cpha to make the data frame format meet the application requirements.
- Step5: Set SPI_CR.spen to 1, and enable the SPI interface.
- Step6: Set SPI_SSN.ssn to 0, make the CS pin output low level to select the slave.
- Step7: Write arbitrary data to SPI_DATA to trigger the host to send SCK.
- Step8: The query waits for SPI_STAT.SPIF to change to 1, and the data sent by the slave has been received.

Page 436

- Step9: Read the received data from SPI_DATA.
- Step10: If the data to be received is not completed, jump to Step7 to continue execution.
- Step11: Set SPI_SSN.ssn to 1, make the CS pin output high level to release the slave.

Notice:

- GPIO can be used instead of CS to realize chip select output, which is mostly used in multi-computer communication systems.
- SPI_SSN.ssn must be set to 0 during the transfer process, and SPI_SSN.ssn must be set to 1 after the transfer is completed.

18.4.3 SPI slave transmission example

- Step1: Map SSN/SCK/MISO/MOSI according to the relevant description of the pin digital multiplexing function in the GPIO chapter To the required pin; and configure the SSN/SCK/MOSI pin as input mode, and configure the MISO pin as output mode Mode. See the GPIO port auxiliary controller for the source of SSN pin.
- Step2: Set SPI_CR.mstr to 0 to make SPI work in slave mode.
- Step3: Configure SPI_CR.cpol and SPI_CR.cpha to make the data frame format meet the application requirements.
- Step4: Set SPI_CR.spen to 1, and enable the SPI interface.

Step5: The query waits for the SSN pin to be pulled low, and the master selects the SPI slave.

Step6: Write the data to be sent into SPI_DATA, and wait for SPIF to become 1.

Step7: Read SPI_DATA to clear the SPIF flag.

Step8: When it is found that the SSN pin is low and the data to be sent has not yet been completed, jump to Step6.

Step9: The query waits for the SSN pin to be pulled high, and the master releases the SPI slave.

18.4.4 SPI slave receiving example

Step1: Map SSN/SCK/MISO/MOSI according to the relevant description of the pin digital multiplexing function in the GPIO chapter

To the required pin; and configure the SSN/SCK/MOSI pin as input mode, and configure the MISO pin as output mode

Mode. See the GPIO port auxiliary controller for the source of SSN pin.

Step2: Set SPI_CR.mstr to 0 to make SPI work in slave mode.

Step3: Configure SPI_CR.cpol and SPI_CR.cpha to make the data frame format meet the application requirements.

Step4: Set SPI_CR.spen to 1, and enable the SPI interface.

Step5: The query waits for the SSN pin to be pulled low, and the master selects the SPI slave.

Page 437

Step6: The query waits for SPI_STAT.SPIF to change to 1, and the data sent by the host has been received.

Step7: Read the received data from SPI_DATA.

Step8: If the data to be received is not completed, jump to Step6 to continue execution.

Step9: The query waits for the SSN pin to be pulled high, and the master releases the SPI slave.

Page 438

18.6 SPI register description

Register list

SPI base address: 0x40000800

Offset	register name	access	Register description
0x00	SPI_CR	RW	SPI configuration register
0x04	SPI_SSN	RW	SPI Chip Select Configuration Register
0x08	SPI_STAT	RO	SPI status register
0x0c	SPI_DATA	RW	SPI data register

Table 18-2 SPI register list

Page 439**18.6.1 SPI Configuration Register (SPI_CR)**

Address offset: 0x00

Reset value: 0x0000 0014

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16
Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Reserved														
spr2 spen														
RW RW														
mstr cpol cpha spr1 spr0														
Res														
RW RW RW RW RW														

Bit	mark	Function description
-----	------	----------------------

31:8	Reserved	
7	spr2	Baud rate selection bit 2 Refer to spr0.
6	spen	SPI module enable control 0-prohibited 1-Enable
5	Reserved	
4	mstr	SPI working mode configuration 0-slave mode 1-Host mode
3	cpol	SCK line idle state configuration 0-low level 1-high level
2	cpha	Clock phase configuration 0-first edge 1-second edge
1	spr1	Baud rate selection bit 1 Reference spr0

Page 440

0	spr0	Baud rate selection bit 0			
		spr2	spr1	spr0	SCK Rate
		0	0	0	PCLK /2
		0	0	1	PCLK /4
		0	1	0	PCLK /8
		0	1	1	PCLK /16
		1	0	0	PCLK /32
		1	0	1	PCLK /64
		1	1	0	PCLK /128
		1	1	1	Reserved

Table 18-3 Host mode baud rate selection

Page 441

18.6.2 SPI Chip Select Configuration Register (**SPI_SSN**)

Address offset: 0x04

Reset value: 0x000000FF

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16
Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
ssn RW														

Bit mark Function description

31:1 Reserved

0 ssn SPI_CS output level configuration in host mode

0: SPI_CS port outputs low level

1: SPI_CS port outputs high level

18.6.3 SPI Status Register (SPI_STAT)

Address offset: 0x08

Reset value: 0x00000004

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16
Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
spif sserr mdf RO Res. RO RO Reserved														

Bit	mark	Function description
31:8	Reserved	
7	spif	Transfer complete flag 1: SPI bus has completed one byte transmission 0: SPI bus is being transmitted
6	Reserved	
5	sserr	Slave mode SSN error flag
4	mdf	In host mode, conflict flag 1: SSN pin level is low 0: SSN pin level is high
3:0	Reserved	

18.6.4 SPI Data Register (SPI_DATA)

Address offset: 0x0c

Reset value: 0x00000000

31	30	29	28	27	26	25	twenty four	twenty five	twenty six	twenty seven	twenty eight	20	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	spdat
Reserved																

Bit	mark	Function description
31:8	Reserved	
7:0	spdat	Data register In the sending mode, write the byte to be sent to this register; In the receiving mode, read the received byte from this register;

Page 444

19 clock calibration module (CLKTRIM)

19.1 Introduction to CLK_TRIM

The CLK_TRIM (Clock Trimming) module is a circuit specially used to calibrate/monitor the clock. In calibration mode

Select an accurate clock source to calibrate the inaccurate clock source, repeat the calibration, and adjust the parameters of the inaccurate clock source

Until the frequency of the calibrated clock source meets the accuracy requirements. There will be a certain error in the count value in the calibration mode.

Within the tolerance of Xu's accuracy. In the monitoring mode, select a stable clock source to monitor the system working clock.

Under the monitoring cycle, monitor whether the working clock of the system fails and generates an interrupt. In calibration mode and

In monitor mode, all required clock sources must be initialized and enabled. For the specific configuration process, please refer to Chapter 4 System Controller.

19.2 Main features of CLK_TRIM

CLK_TRIM supports the following features:

Calibration mode

Monitoring mode

32-bit reference clock counter can be loaded with initial value

32-bit clock counter to be calibrated with configurable overflow value

6 reference clock sources

4 clock sources to be calibrated

Support interrupt mode

Page 445

19.3 CLK_TRIM function description

19.3.1 CLK_TRIM calibration mode

The calibration mode is mainly used to select an accurate clock source as the reference clock to calibrate an inaccurate time to be calibrated
Zhong Yuan.

The software is repeatedly calibrated according to the following operating procedure, adjusting the parameters of the clock source to be calibrated until
Meet the frequency accuracy requirements.

19.3.1.1 Operation Process

1. Set the CLKTRIM_CR.refclk_sel register to select the reference clock.
2. Set the CLKTRIM_CR.calclk_sel register to select the clock to be calibrated.
3. Set the CLKTRIM_REFCON.rcntval register as the calibration time.
4. Set the CLKTRIM_CR.IE register to enable interrupts.
5. Set the CLKTRIM_CR.trim_start register to start calibration.
6. The reference clock counter and the clock counter to be calibrated start counting.
7. When the reference clock counter counts down from the initial value to 0, CLKTRIM_IFR.stop is set to 1, triggering an interrupt.
8. The interrupt service subroutine judges that CLKTRIM_IFR.stop is 1, read the register CLKTRIM_REFCNT
And the value of CLKTRIM_CALCNT,
9. Clear the CLKTRIM_CR.trim_start register to end the calibration.

Notice:

– During the calibration process, the calibration mode may cause the clock counter to be calibrated when the calibration time is too long.

In the case of overflow before CLKTRIM_IFR.stop is set to 1, CLKTRIM_IFR.calcnt_of is set to 1, trigger
Interrupted. When the interrupt service routine finds that CLKTRIM_IFR.calcnt_of is set to 1, clear it
The CLKTRIM_CR.trim_start register ends the calibration.

In this case, the calibration cannot be performed correctly, and the calibration time must be adjusted and recalibrated.

The specific steps are:

Set the CLKTRIM_REFCON .rcntval register to adjust the calibration time.

Set the CLKTRIM_CR .trim_start register to restart calibration.

Page 446

19.3.2 CLK_TRIM monitoring mode

The monitoring mode is mainly used to select a stable clock source as the reference clock, and monitor the system under the set time period.

Abnormal state of the system working clock. In monitoring mode, only external XTH clock or external XTL clock can be selected

As the monitored clock.

19.3.2.1 Operation Process

1. Set the CLKTRIM_CR.refclk_sel register to select the reference clock.
2. Set the CLKTRIM_CR.calclk_sel register to select the monitored clock.
3. Set the CLKTRIM_REFCON .rcntval register to monitor interval time.
4. Set the CLKTRIM_CALCON. ccntval register to the overflow time of the monitored clock counter.
5. Set the CLKTRIM_CR.mon_en register to enable the monitoring function.
6. Set the CLKTRIM_CR.IE register to enable interrupts.
7. Set the CLKTRIM_CR.trim_start register to start monitoring.
8. The reference clock counter and the monitored clock counter start counting.
9. When the count of the reference clock counter reaches the monitoring interval time, judge whether the monitored clock counter overflows.
If it overflows, it means that the monitored clock is working normally. If there is no overflow, the monitored clock is invalid,
CLKTRIM_IFR .xtal32k_fault/xtal32m_fault is set to 1 to trigger an interrupt.
10. Handle the interrupt service subroutine and clear the interrupt flag bit
CLKTRIM_IFR .xtal32k_fault/xtal32m_fault, clear the CLKTRIM_CR .trim_start register
End monitoring.

19.4 CLK_TRIM register description

Register list		Base address: 0x40001800	
Offset	register name	Access	register description
0x00	CLKTRIM_CR	RW	Configuration register.
0x04	CLKTRIM_REFCON	RW	Refer to the counter initial value configuration register.
0x08	CLKTRIM_REFCNT	RO	Refer to the counter value register.
0x0c	CLKTRIM_CALCNT	RO	Calibration counter value register.
0x10	CLKTRIM_IFR	RO	Interrupt flag bit register.
0x14	CLKTRIM_ICLR	RW	Interrupt flag clear register
0x18	CLKTRIM_CALCON	RW	Calibration counter overflow value configuration register

Table 19-1 Register List

19.4.1 Configuration Register (CLKTRIM_CR)

Offset address: 0x00
 Reset value: 0x0000 0000

31 30 29	28 27	26 25 24 23 22 21 20 19 18 17 16	Reserved												
15 14 13	12 11	10 9 8 7 6 5 4 3 2 1 0	Reserved	IE mon	calclk_sel	refclk_sel	trim_start								
				_en				RW							

Bit	mark	Function description
31:8	Reserved	
7	IE	Interrupt enable register 0-disabled 1-Enable
6	mon_en	Monitor mode enable register 0-disabled 1-Enable
5:4	calclk_sel	to be calibrated/monitored clock selection register
	00 ----	RCH
	01 ----	XTH
	10 ----	RCL
	11 ----	XTL
3:1	refclk_sel	reference clock selection register
	000 ----	RCH
	001 ----	XTH
	010 ----	RCL
	011 ----	XTL
	100 ----	IRC10K
	101 ----	EXT_CLK_IN
0	trim_start	Calibration/monitoring start register 0-stop 1-Start

19.4.2 Reference Counter Initial Value Configuration Register (**CLKTRIM_REFCON**)

Offset address: 0x04

Reset value: 0x00000000

31 30 29	28 27	26 25 24 23 22 21 20 19 18 17 16	rcntval[31:16]												
15 14 13	12 11	10 9 8 7 6 5 4 3 2 1 0	RW												

rcntval[15:0]

RW

Bit	mark	Function description
31:0	rcntval	Reference counter initial value

19.4.3 Reference Counter Value Register (CLKTRIM_REFCNT**)**

Offset address: 0x08

Reset value: 0x00000000

31 30 29	28 27	26 25 24 23 22 21 20 19 18 17 16
refcnt[31:16]		
RO		
15 14 13	12 11	10 9 8 7 6 5 4 3 2 1 0
refcnt[15:0]		
RO		

Bit	mark	Function description
31:0	refcnt	Reference counter value

19.4.4 Calibration Counter Value Register (CLKTRIM_CALCNT**)**

Offset address: 0x0

Reset value: 0x00000000

31 30 29	28 27	26 25 24 23 22 21 20 19 18 17 16
calcnt[31:16]		
RO		
15 14 13	12 11	10 9 8 7 6 5 4 3 2 1 0
calcnt[15:0]		
RO		

Bit	mark	Function description
31:0	calcnt	Calibration counter value

HC32L110 Series User Manual Rev2.31

Page 450 of 527

Page 451

19.4.5 Interrupt Flag Bit Register (CLKTRIM_IFR)

Offset address: 0x10

Reset value: 0x0000 0000

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16

18 17 16

10 9

Reserved

Kō Kō Kō Kō

Bit mark Function description

31:4 Reserved

3 xth_fault XTH failure flag

CLKTRIM_ICLR.xth_fault_clr Write zero to clear this flag.

2 xtl_fault XTL failure flag

CLKTRIM_ICLR.xtl_fault_clr Write zero to clear this flag

1 calcnt_of Calibration counter overflow flag

CLKTRIM CR.start write zero to clear this flag

0	stop	Refer to the counter stop sign. CLKTRIM_CR.start writes zero to clear this flag
---	------	--

Page 452**19.4.6 Interrupt Flag Bit Clear Register (CLKTRIM_ICLR)**

Offset address: 0x14

Reset value: 0xf

31 30 29 28 27 26 25 24 23 22 21 20 19	18 17 16												
Reserved													
15 14 13 12 11	10 9 8 7 6 5 4 3 2 1 0												
	Reserved				xth_f	xtl_f	Reserved						
					ault_	ault_							
					clr	clr							
					RW	RW							

Bit	mark	Function description
31:4	Reserved	
3	xth_fault_clr	Clear XTH failure flag, write zero to clear.
2	xtl_fault_clr	Clear the XTL failure flag, write zero to clear.
1:0	Reserved	

Page 453**19.4.7 Calibration counter overflow value configuration register (**CLKTRIM_CALCON**)**

Offset address: 0x18

Reset value: 0xffff ffff

31 30 29	28 27	26 25 24 23 22 21 20 19 18 17 16	ccntval[31:16]	RW
15 14 13	12 11	10 9 8 7 6 5 4 3 2 1 0	ccntval[15:0]	RW

Bit	mark	Function description
31:0	ccntval	Calibration counter overflow value

20 cyclic redundancy check (CRC)

20.1 Overview

Cyclic Redundancy Check (CRC) calculation unit takes data stream or data block as input, under the control of generator polynomial Generate an output number. This output number is often used to verify the correctness and completeness of data transmission or storage. This module supports calculating CRC value and checking CRC value.

20.2 Main features

- An implementation standard: ISO/IEC13239
- One encoding method: CRC-16, $x^{16} + x^{12} + x^5 + 1$
- Three kinds of writing bit width: 8bit, 16bit, 32bit
- Two working modes: CRC encoding mode, CRC check mode

20.3 Functional description

20.3.1 Working Mode

This module supports two working modes: CRC encoding mode and CRC check mode.
The CRC encoding mode is to input a certain amount of raw data to the CRC module and obtain the output generated by the CRC module Value (CRC_RESULT.RESULT). The CRC check mode is to input a certain amount of raw data to the CRC module According to the +CRC check value, verify whether the original data matches the CRC check value (CRC_RESULT.FLAG).

20.3.2 Encoding method

This module supports CRC-16 encoding, the calculation result is 16 bits, and the generator polynomial is $x^{16} + x^{12} + x^5 + 1$.

20.3.3 Write bit width

This module supports three write bit widths: 8bit, 16bit, and 32bit. Writes with different bit widths need to comply with the "bit width consistent, The principle of "low first, then high", that is, "every time data is written, it must be written to a register with the same valid data bit width as this time In the memory, and the lower data is written before the higher data".
The following shows how to write the same sequence of data with three bit widths, and the output results are the same.

8bit bit width write: 0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77

16bit bit width write: 0x1100, 0x3322, 0x5544, 0x7766

32bit bit width write: 0x33221100, 0x77665544

20.4 Programming example

20.4.1 CRC-16 encoding mode

Step 1: Write 0xFFFF to CRC_RESULT to initialize CRC calculation.

Step 2: Write the original data to be encoded into the CRC_DATA register in sequence, and the write bit width can be 8bit, 16bit, 32bit.

Step 3: Read CRC_RESULT.RESULT to get the CRC value.

20.4.2 CRC-16 check mode

Step 1: Write 0xFFFF to CRC_RESULT to initialize CRC calculation.

Step 2: Write the coded data sequence to the CRC_DATA register in turn, the write bit width can be 8bit, 16bit, 32bit.

Step 3: The value of CRC_RESULT.FLAG determines whether the encoded data sequence has been tampered with.

20.5 Register description

20.5.1 Register List

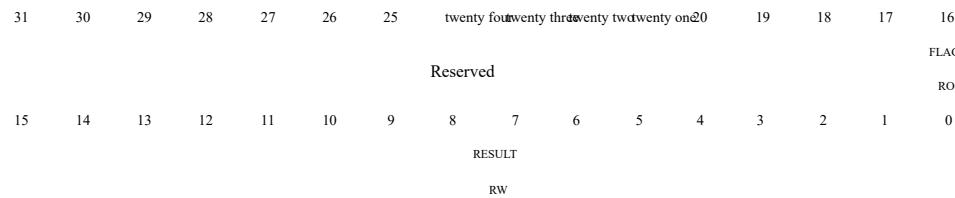
Base address: 0x4002 0900

register	Offset address	describe
CRC_RESULT	0x04	CRC result register
CRC_DATA	0x80	CRC data register

Page 457**20.5.2 Result Register (CRC_RESULT)**

Offset address: 0x04

Reset value: 0x0000 0000



Bit	symbol	describe
16	FLAG	CRC check result 0: current CRC check error 1: The current CRC check is correct
15:0	RESULT	CRC calculation result

Read this register to get the CRC calculation result
Write 0xFFFF to this register to initialize the CRC calculation

20.5.3 Data Register (**CRC_DATA**)

Offset address: 0x80

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA[31:16]															
WO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA[15:0]															
WO															

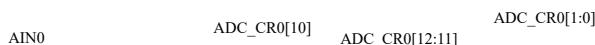
Bit	symbol	Function description
31:0	DATA	This register is used to write data that needs to be calculated, and supports 3 types of write bit widths 8bit writing method: * ((uint8_t *)0x40020980) = 0XX 16bit writing method: * ((uint16_t *)0x40020980) = 0XXXX 32bit writing method: * ((uint32_t *)0x40020980) = 0XXXXXXXX

21 Analog-to-digital converter (**ADC**)

21.1 Introduction to Module

The external analog signal needs to be converted into a digital signal before it can be further processed by the MCU. This series integrates a A 12-bit high-precision, high conversion rate successive approximation analog-to-digital converter (SAR ADC) module. Has the following characteristics:
 12-bit conversion accuracy;
 1Msps conversion speed (VCC>2.7V);
 12 conversion channels: 9 pin channels, built-in temperature sensor, built-in 1.2v reference voltage, 1/3 power supply
 Pressure
 4 kinds of reference sources: power supply voltage, ExRef pin, built-in 1.5v reference voltage, built-in 2.5v reference voltage;
 ADC voltage input range: 0~Vref;
 3 conversion modes: single conversion, continuous conversion, cumulative conversion;
 The software can configure the conversion rate of ADC;
 Built-in signal amplifier can convert high-impedance signals;
 Support on-chip peripherals to automatically trigger ADC conversion, effectively reducing chip power consumption and improving the real-time performance.

21.2 ADC block diagram



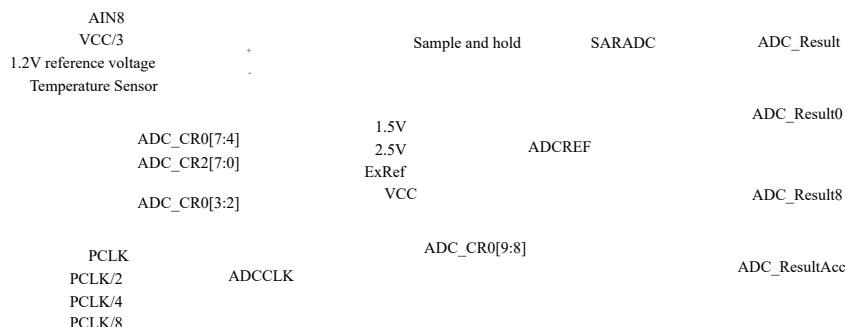


Figure 21-1 ADC schematic block diagram

Page 459**21.3 Conversion timing and conversion speed**

The ADC conversion sequence is shown in the figure below: A complete ADC conversion consists of a sampling process and a successive comparison. Among them, the sampling process requires 4~12 AdcClks, which are configured by ADC_CR0.SAM; the successive comparison process requires 16 AdcClk. Therefore, a total of 20~28 AdcClks are required for one ADC conversion.

The unit of ADC conversion speed is sps, that is, how many ADC conversions are performed per second. Calculation method of ADC conversion speed: The method is: frequency of AdcClk/number of AdcClk required for one ADC conversion.



Figure 21-2 ADC conversion timing diagram

ADC conversion speed is related to ADC reference voltage and VCC voltage. The maximum conversion speed is shown in the following table:

ADC reference voltage	VCC voltage	Maximum conversion speed	Maximum AdcClk frequency
Internal 1.5V	1.8~5.5V	200Ksps	4MHz
Internal 2.5V	2.8~5.5V	200Ksps	4MHz
VCC / ExRef	1.8~2.4V	200Ksps	4MHz
VCC / ExRef	2.4~2.7V	500Ksps	16MHz
VCC / ExRef	2.7~5.5V	1Msps	24MHz

Page 460**21.4 Single conversion mode**

In single conversion mode, only one conversion is performed after the ADC is started, and all 12 ADC channels can be converted. Change. This mode can either be started by setting the ADC_CR0.START bit or by setting the ADC_CR1[9:0] external Trigger start. Once the ADC conversion of the selected channel is completed, the ADC_CR0.START bit is automatically cleared and the conversion is The result is stored in the ADC_result register.

Start the **ADC** single conversion operation flow through the **START** bit :

Step1: Configure the corresponding bits of P0ADS~P3ADS, and configure the ADC channel to be converted as an analog port.

Step2: Set P3ADS.6 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the *ADC* reference voltage does not select the external reference voltage pin, you can skip this step.

Step3: Set BGR_CR.BGR_EN to 1, enable the BGR module.

Step4: Set ADC_CR0.ADCEN to 1 to enable the ADC module.

Step5: Delay 20uS, and wait for the ADC and BGR modules to start up.

Step6: Set ADC_CR1.CT to 0 and select single conversion mode.

Step7: Configure ADC_CR0. SREF, select ADC reference voltage.

Step8: Configure ADC_CR0. SAM and ADC_CR0. CLKSEL to set the conversion speed of ADC.

Step9: Configure ADC_CR0. SEL, select the channel to be converted.

Step10: Set ADC_CR0.START to 1, start ADC single conversion.

Step11: Wait for ADC_CR0.START to change to 0, read the ADC_result register to get the ADC conversion result
fruit.

Step12: If you need to convert other channels, repeat Step9~Step11.

Step13: Set ADC_CR0.ADCEN and BGR_CR.BGR_EN to 0, turn off the ADC module and BGR module
Piece.

Start **ADC** single conversion operation flow through external trigger :

Step1: Configure the corresponding bits of P0ADS~P3ADS, and configure the ADC channel to be converted as an analog port.

Step2: Set P3ADS.6 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the *ADC* reference voltage does not select the external reference voltage pin, you can skip this step.

Step3: Set BGR_CR.BGR_EN to 1, enable the BGR module.

Step4: Set ADC_CR0.ADCEN to 1 to enable the ADC module.

Step5: Delay 20uS, and wait for the ADC and BGR modules to start up.

Step6: Set ADC_CR1.CT to 0 and select single conversion mode.

Step7: Set ADC_HT to 0x00.

Step8: Set ADC_CR1.HtCmp to 1, enable ADC high threshold comparison function.

Step9: Set ADC_CR0.IE to 1, enable ADC interrupt.

Step10: Enable ADC interrupt in NVIC interrupt vector table.

Step11: Configure ADC_CR0.SREF, select ADC reference voltage.

Step12: Configure ADC_CR0.SAM and ADC_CR0.CLKSEL to set the conversion speed of ADC.

Step13: Configure ADC_CR0.SEL and select the channel to be converted.

Step14: Set ADC_IFR to 0x00 and clear the ADC interrupt flag.

Step15: Configure ADC_CR1.TRIGS1 and ADC_CR1.TRIGS0 to select external trigger conditions.

Step16: When the external trigger condition triggers the ADC to complete the conversion, the ADC module will generate an interrupt. Users can access the ADC_result register in the interrupt service routine to obtain the ADC conversion result.

Step17: If you need to convert other channels, repeat Step13~Step16.

Step18: Set ADC_CR0.ADCEN and BGR_CR.BGR_EN to 0, turn off the ADC module and BGR module piece.

21.5 Continuous Conversion Mode

In the continuous conversion mode, the ADC can be started once to perform multiple conversions on multiple channels in sequence; convertible ADC The channels are AIN0~AIN7. The total number of ADC conversions is configured by ADCCR2.ADCCNT; The channel is configured by ADC_CR2[7:0]. This mode can either be started by setting the ADC_CR0.START bit or Start by setting the external trigger of ADC_CR2[9:0]. After starting continuous conversion, the ADC module converts sequentially The channels to be converted in AIN0~AIN7 until the total conversion times are completed. After the ADC module completes the total number of conv ADC_IFR. The CONT_INTF bit will be automatically set to 1, and the conversion result will be saved in the corresponding conversion channel ADC_result0~ADC_result7 registers. If the total number of conversions is greater than the number of ADC channels to be converted, Then only the last conversion result is saved in the ADC_result0~ADC_result7 registers.

The following figure demonstrates the process of 10 consecutive conversions on AIN0, AIN1, and AIN5. START through the register

After it is set to 1, the internal state of the ADC will convert AIN0, AIN1, and AIN5 in sequence until ADCCNT

The count value becomes 0.

ADCEN										
START										
ADCCNT	09	08	07	06	05	04	03	02	01	00
ADCCHx	00	01	05	00	01	05	00	01	05	00
CONT_INTF										
ADC_Result0										
ADC_Result1										
ADC_Result5										

Figure 21-3 Example of ADC continuous conversion process

Start the ADC continuous conversion operation flow through the START bit :

Step1: Configure the corresponding bits of P0ADS~P3ADS, and configure the ADC channel to be converted as an analog port.

Step2: Set P3ADS.6 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the ADC reference voltage does not select the external reference voltage pin, you can skip this step.

Step3: Set BGR_CR.BGR_EN to 1, enable the BGR module.

Step4: Set ADC_CR0.ADCEN to 1 to enable the ADC module.

Step5: Delay 20uS, and wait for the ADC and BGR modules to start up.

Step6: Set ADC_CR1.CT to 1, select continuous conversion mode.

Step7: Set ADC_CR1[14:12] to 0, turn off the conversion result comparison function.

Step8: Configure ADC_CR2. ADCCNT, select the total number of consecutive conversions.

Step9: Configure ADC_CR0. SREF, select ADC reference voltage.

Step10: Configure ADC_CR0. SAM and ADC_CR0. CLKSEL to set the conversion speed of ADC.

Step11: Configure ADC_CR2[7:0] and select the channel to be converted.

Step12: Set ADC_ICLR. CONT_INTC to 0, clear ADC_IFR. CONT_INTF flag.

Step13: Set ADC_CR0. StateRst to 1, reset the continuous conversion state.

Step14: Set ADC_CR0.START to 1, start ADC continuous conversion.

Step15: Wait for ADC_IFR.CONT_INTF to change to 1, read ADC_result0~ADC_result7 registers

To obtain the conversion result of the corresponding channel.

Step16: If you need to convert other channels, repeat Step11~Step15.

Step17: Set ADC_CR0.ADCEN and BGR_CR.BGR_EN to 0, turn off the ADC module and BGR module

Piece.

21.6 Continuous conversion accumulation mode

In the continuous conversion accumulation mode, the ADC can be started once to perform multiple conversions on multiple channels and the value of e

The result is accumulated; the ADC channels that can be converted are AIN0~AIN7. The total number of ADC conversions is determined by

ADCCR2.ADCCNT is configured; the channel to be converted is configured by ADC_CR2[7:0]. This mode is both

It can be started by setting the ADC_CR0.START bit or by setting the external trigger of ADC_CR2[9:0].

After the continuous conversion is started, the ADC module sequentially converts the channels to be converted in AIN0~AIN7 until the total number o

become. After the ADC module completes the total number of conversions, the ADC_IFR.CONT_INTF bit will be automatically set to 1, and the con

The accumulated value is stored in the ADC_result_acc register.

The following figure demonstrates the process of 10 consecutive conversions and accumulation of AIN0, AIN1, and AIN5. Through the register

After START is set to 1, the internal state of the ADC will convert AIN0, AIN1, and AIN5 in turn until

The count value of ADCCNT becomes 0. Each time the conversion is completed, the ADC_result_acc register will automatically accumulate

add. The conversion results of AIN0, AIN1, and AIN5 given in the figure are 0x010, 0x020, and 0x040 in sequence.

ADCEN										
START										
ADCCNT	09	08	07	06	05	04	03	02	01	00
ADCCHx	00	01	05	00	01	05	00	01	05	00
CONT_INTF										
ADC_ResultAcc	000	010	030	070	080	0A0	0E0	0F0	110	150
										160

Figure 21-4 ADC continuous conversion and accumulation process example

Start the **ADC** continuous conversion and accumulation operation flow through the **START** bit :

Step1: Configure the corresponding bits of P0ADS~P3ADS, and configure the ADC channel to be converted as an analog port.

Step2: Set P3ADS.6 to 1, and configure the ADC external reference voltage pin as an analog port.

Note: If the *ADC* reference voltage does not select the external reference voltage pin, you can skip this step.

Step3: Set BGR_CR.BGR_EN to 1, enable the BGR module.

Step4: Set ADC_CR0.ADCEN to 1 to enable the ADC module.

Step5: Delay 20uS, and wait for the ADC and BGR modules to start up.

Step6: Set ADC_CR1.CT to 1, select continuous conversion mode.

Step7: Set ADC_CR1[14:12] to 0, turn off the conversion result comparison function.

Step8: Set ADC_CR1.RACC_EN to 1, to enable ADC conversion automatic accumulation function.

Step9: Configure ADC_CR2.ADCCNT, select the total number of consecutive conversions.

Step10: Configure ADC_CR0.SREF, select ADC reference voltage.

Step11: Configure ADC_CR0.SAM and ADC_CR0.CLKSEL to set the conversion speed of ADC.

Step12: Configure ADC_CR2[7:0] and select the channel to be converted.

Step13: Set ADC_ICLR.CONT_INTC to 0 and clear ADC_IFR.CONT_INTF flag.

Step14: Set ADC_CR1.RACC_CLR to 0, clear ADC_result_acc register.

Step15: Set ADC_CR0.StateRst to 1, reset the continuous conversion state.

Step16: Set ADC_CR0.START to 1, start ADC continuous conversion.

Step17: Wait for ADC_IFR.CONT_INTF to become 1, read ADC_result_acc register to obtain conversion

The cumulative value of the result.

Step18: If you need to convert other channels, repeat Step12~Step17.

Step19: Set ADC_CR0.ADCEN and BGR_CR.BGR_EN to 0, turn off the ADC module and BGR module

Piece.

Page 466**21.7 Comparison of ADC conversion results**

When the ADC conversion is completed, the ADC conversion result can be compared with the threshold set by the user, and the upper threshold comparison function needs to be set.

Lower threshold value comparison, interval value comparison. This function needs to set the corresponding control bits HtCmp, LtCmp, RegCmp.

1. This function can realize the automatic monitoring of the analog quantity, until the ADC conversion result meets the user's expectations.

Discontinue application user program access.

Upper threshold comparison: ADC_IFR. HHT_INTF when the ADC conversion result is within the range of [ADC_HT, 4095]

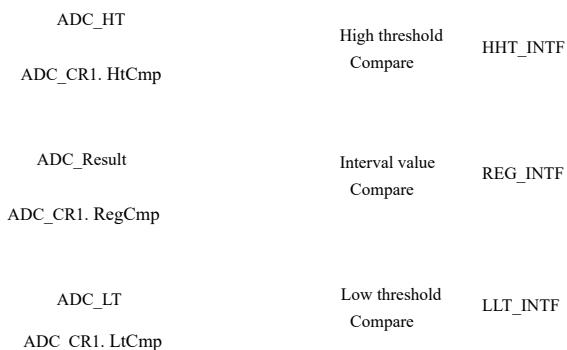
Set to 1; write 0 to ADC_ICLR. HHT_INTC to clear ADC_IFR. HHT_INTF.

Lower threshold comparison: when the ADC conversion result is in the [0, ADC_LT) interval, ADC_IFR. LLT_INTF is set to 1;

Write 0 to ADC_ICLR. LLT_INTC to clear ADC_IFR. LLT_INTF.

Interval value comparison: When the ADC conversion result is within the [ADC_LT, ADC_HT) interval, then ADC_IFR.

Set REG_INTF to 1; write 0 to ADC_ICLR. REG_INTC to clear ADC_IFR. REG_INTF.



Page 467**21.8 ADC Interrupt**

The ADC interrupt request is shown in the following table:

Interrupt source	Interrupt flag	Interrupt enable
ADC continuous conversion completed	ADC_IFR.CONT_INTF	
ADC conversion result is located in the internal memory	ADC_IIRREG_INTF	
ADC conversion result is in the upper threshold	ADC_HTR.HHT_INTF	ADC_CR0.IE
ADC conversion result comparison lower threshold	ADC_LTR.LLT_INTF	

21.9 Use a temperature sensor to measure the ambient temperature

The output voltage of the temperature sensor will change with the change of the ambient temperature, so according to the output voltage of the temperature sensor, the corresponding ambient temperature can be calculated. When the measurement channel of the ADC module selects the output voltage of the temperature sensor, the ambient temperature can be measured.

Calculated as follows:

$$\text{Ambient temperature} = 25 + 0.0839 \times V_{ref} \times (\text{AdcValue} - \text{Trim})$$

Among them: V_{ref} is the reference voltage of the current ADC module, and the value is 1.5 or 2.5.

AdcValue is the result of the ADC module measuring the output voltage of the temperature sensor, and the value is 0~4095.

Trim is a 16-bit calibration value, which needs to be read from the Flash memory during calculation. The storage address is detailed below the surface.

ADC reference voltage Calibration value storage address Calibration accuracy

Internal 1.5V	0x00100C34	$\pm 5^{\circ}\text{C}$
Internal 2.5V	0x00100C36	$\pm 5^{\circ}\text{C}$

The calculation example is as follows:

Condition 1: $V_{ref}=2.5$, $\text{AdcValue}=0x7E5$, $\text{Trim}=0x76C$:

$$\text{Temperature 1: } 25 + 0.0839 \times 2.5 \times (0x7E5 - 0x76C) = 50^{\circ}\text{C}.$$

Condition 2: $V_{ref}=1.5$, $\text{AdcValue}=0x72D$, $\text{Trim}=0x76C$:

$$\text{Temperature 3: } 25 + 0.0839 \times 1.5 \times (0x72D - 0x76C) = 17^{\circ}\text{C}.$$

Page 468

Operation process of measuring ambient temperature through **ADC** :

Step1: Set BGR_CR.BGR_EN to 3, enable BGR module and temperature sensor module.

Step2: Set ADC_CR0.ADCEN to 1 to enable the ADC module.

Step3: Delay 20uS, and wait for the ADC and BGR modules to start up.

Step4: Set ADC_CR1.CT to 0 and select single conversion mode.

Step5: Configure ADC_CR0.SREF, select ADC reference voltage as internal 1.5V or internal 2.5V.

Step6: Configure ADC_CR0.SAM and ADC_CR0.CLKSEL to set the conversion speed of ADC.

Step7: Set ADC_CR0.SEL to 0x0A, and select the channel to be converted as the output of the temperature sensor.

Step8: Set ADC_CR0.BUFEN to 1, enable the input signal amplifier.

Step9: Set ADC_CR0.START to 1, start ADC single conversion.

Step10: Wait for ADC_CR0.START to become 0, read the ADC_result register to get the ADC conversion result
fruit.

Step11: Set ADC_CR0.ADCEN and BGR_CR.BGR_EN to 0, turn off the ADC module and BGR module
Piece.

Step12: Read the calibration value of the temperature sensor, and calculate the current ambient temperature according to the formula.

Page 469

21.10 ADC module registers

Base address 0x40002400

register	Offset address	describe
ADC_CR0	0x004	ADC configuration register 0
ADC_CR1	0x008	ADC Configuration Register 1
ADC_CR2	0x00C	ADC Configuration Register 2
ADC_result0	0x030	ADC channel 0 conversion result
ADC_result1	0x034	ADC channel 1 conversion result
ADC_result2	0x038	ADC channel 2 conversion result
ADC_result3	0x03C	ADC channel 3 conversion result
ADC_result4	0x040	ADC channel 4 conversion result
ADC_result5	0x044	ADC channel 5 conversion result
ADC_result6	0x048	ADC channel 6 conversion result
ADC_result7	0x04C	ADC channel 7 conversion result
ADC_result8	0x050	ADC channel 8 conversion result
ADC_result_acc	0x054	Accumulated value of ADC conversion result
ADC_HT	0x058	ADC compare upper threshold
ADC_LT	0x05C	ADC compare lower threshold
ADC_IFR	0x060	ADC interrupt flag register
ADC_ICLR	0x064	ADC interrupt clear register
ADC_result	0x068	ADC conversion result

Table 21-1 ADC Register

21.10.1 ADC Configuration Register 0 (ADC_CR0)

Offset address 0x004

Reset value 0x000013F0

31	30	29	28	27	26	25	twent	four	twent	twent	on	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

State Rst	IE Res.	SAM	BUF EN	SREF	SEL	CLKSEL	STA RT	ADC EN
R/W/R/W		R/W	R/W	R/W	R/W		R/W	R/W/R/W
Bit	mark	Function description						
31:16	Reserved	Reserve						
15	StateRst	ADC continuous conversion status control						
		1: Reset ADC continuous conversion status						
		0: invalid						
14	IE	ADC interrupt control						
		1: Enable interrupt						
		0: Disable interrupt						
13	Reserved	Reserve						
12:11	SAM	ADC sampling period selection						
		00: 4 sampling periods						
		01: 6 sampling periods						
		10: 8 sampling periods						
		11: 12 sampling periods						
10	BUFEN	ADC input signal amplifier control						
		0: Turn off the amplifier, and the external input signal is directly connected to the ADC.						
		1: Turn on the amplifier, and the external input signal is amplified by the amplifier and connected to the ADC for high-impedance signals.						
9:8	SREF	ADC reference voltage selection						
		00: Internal 1.5V						
		01: Internal 2.5V						
		10: External reference voltage ExRef (P3.6)						
		11: Power supply voltage						
7:4	SEL	ADC conversion channel selection (single conversion mode)						
		0000: Select channel 0 to input P2.4						
		0001: Select channel 1 to input P2.6						
		0010: Select channel 2 to input P3.2						
		0011: Select channel 3 to input P3.3						
		0100: Select channel 4 to input P3.4						

3:2	CLKSEL	0101: Select channel 5 to input P3.5 0110: Select channel 6 to input P3.6 0111: Select channel 7 to input P0.1 1000: select channel 8 input P0.2 1001: VCC/3	Note: ADC_CR0.BUFEN must be 1
1	START	1010: Built-in temperature sensor output voltage Note: ADC_CR0.BUFEN must be 1 1011: Internal reference 1.2V output voltage Note: ADC_CR0.BUFEN must be 1	
0	ADCEN	ADC clock selection 00: PCLK clock 01: PCLK clock divided by 2 10: PCLK clock divided by 4 11: PCLK clock divided by 8	
		ADC conversion control 1: Start ADC conversion 0: Stop ADC conversion	
		ADC enable control 1: Enable ADC 0: Disable ADC	

Page 472**21.10.2 ADC Configuration Register 1 (ADC_CR1)**

Offset address 0x008

Reset value 0x00007000

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16
Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
RACC _CLR	RegCm p	HtCmp	LtCmp	HtCmp	CT									TRIGS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit mark Function description

31:16 Reserved Reserve

ADC conversion result accumulation register is cleared

15 RACC_CLR

1: No effect;

0: The ADC conversion result accumulation register (ADC_result_acc) is cleared.

Note: This bit is read as 0, so you need to pay special attention to the value of this bit when operating this register to prevent misoperation.

ADC interval comparison control

14 RegCmp

1: Enable interval comparison

0: prohibit interval comparison

ADC high threshold comparison control

13 HtCmp

1: Enable high threshold comparison

0: prohibit high threshold comparison

ADC low threshold comparison control

12	LtCmp	1: Enable low threshold comparison 0: prohibit low threshold comparison
11	RACC_EN	ADC conversion result automatic accumulation control 1: Enable the automatic accumulation function of ADC conversion results 0: Disable the automatic accumulation function of ADC conversion results
10	CT	ADC conversion mode selection 1: Continuous conversion mode 0: Single conversion mode
		ADC conversion automatic trigger option 2: 00000: Disable automatic triggering of ADC conversion 00001: Timer0 interrupt, automatically trigger ADC conversion
9:5	TRIGS1	00010: Timer1 interrupt, automatically trigger ADC conversion 00011: Timer2 interrupt, automatically trigger ADC conversion 00100: LPTimer interrupt, automatically trigger ADC conversion 00101: Timer4 interrupt, automatically trigger ADC conversion

Page 473

00110: Timer5 interrupt, automatically trigger ADC conversion
 00111: Timer6 interrupt, automatically trigger ADC conversion
 01000: UART0 interrupt, automatically trigger ADC conversion
 01001: UART1 interrupt, automatically trigger ADC conversion
 01010: LPUART interrupt, automatically trigger ADC conversion
 01011: VC0 interrupt, automatically trigger ADC conversion
 01100: VC1 interrupt, automatically trigger ADC conversion
 01101: RTC interrupt, automatically trigger ADC conversion
 01110: PCA interrupt, automatically trigger ADC conversion
 01111: SPI interrupt, automatically trigger ADC conversion
 10000: P01 interrupt, automatically trigger ADC conversion
 10001: P02 interrupt, automatically trigger ADC conversion
 10010: P03 interrupt, automatically trigger ADC conversion
 10011: P14 interrupt, automatically trigger ADC conversion
 10100: P15 interrupt, automatically trigger ADC conversion
 10101: P23 interrupt, automatically trigger ADC conversion
 10110: P24 interrupt, automatically trigger ADC conversion
 10111: P25 interrupt, automatically trigger ADC conversion
 11000: P26 interrupt, automatically trigger ADC conversion
 11001: P27 interrupt, automatically trigger ADC conversion
 11010: P31 interrupt, automatically trigger ADC conversion
 11011: P32 interrupt, automatically trigger ADC conversion
 11100: P33 interrupt, automatically trigger ADC conversion
 11101: P34 interrupt, automatically trigger ADC conversion
 11110: P35 interrupt, automatically trigger ADC conversion
 11111: P36 interrupt, automatically trigger ADC conversion

Note:

- 1) The TIM4/5/6 interrupt triggers the automatic conversion of ADC, except for the corresponding interrupt of TIM4/5/6 that needs to be enabled.
It is also necessary to configure the spread spectrum and interrupt trigger selection register TIMX_CR of the Advanced Timer.

The interrupt source that triggered the ADC.

- 2) The rising edge of each interrupt flag bit is used to trigger the ADC. If you need to trigger repeatedly, you need to clear Interrupt flag. If you do not need to enter the interrupt service routine, please do not enable the interrupt enable of the NVIC.

ADC conversion automatic trigger option 2:

00000: Disable automatic triggering of ADC conversion
 00001: Select Timer0 interrupt to automatically trigger ADC conversion

4:0 TRIGS0

- 00010: Select Timer1 interrupt to automatically trigger ADC conversion
- 00011: Select Timer2 interrupt to automatically trigger ADC conversion
- 00100: Select the LPTimer interrupt to automatically trigger ADC conversion
- 00101: Select Timer4 interrupt to automatically trigger ADC conversion
- 00110: Select Timer5 interrupt to automatically trigger ADC conversion
- 00111: Select Timer6 interrupt to automatically trigger ADC conversion
- 01000: select UART0 interrupt, automatically trigger ADC conversion

Page 474

- 01001: select UART1 interrupt, automatically trigger ADC conversion
- 01010: select LPUART interrupt, automatically trigger ADC conversion
- 01011: select VC0 interrupt, automatically trigger ADC conversion
- 01100: select VC1 interrupt, automatically trigger ADC conversion
- 01101: Select RTC interrupt and automatically trigger ADC conversion
- 01110: select PCA interrupt, automatically trigger ADC conversion
- 01111: select SPI interrupt, automatically trigger ADC conversion
- 10000: Select P01 interrupt to automatically trigger ADC conversion
- 10001: Select P02 interrupt to automatically trigger ADC conversion
- 10010: Select P03 interrupt to automatically trigger ADC conversion
- 10011: Select P14 interrupt to automatically trigger ADC conversion
- 10100: Select P15 interrupt to automatically trigger ADC conversion
- 10101: Select P23 interrupt to automatically trigger ADC conversion
- 10110: Select P24 interrupt to automatically trigger ADC conversion
- 10111: Select P25 interrupt to automatically trigger ADC conversion
- 11000: Select P26 interrupt to automatically trigger ADC conversion
- 11001: Select P27 interrupt to automatically trigger ADC conversion
- 11010: Select P31 interrupt to automatically trigger ADC conversion
- 11011: Select P32 interrupt to automatically trigger ADC conversion
- 11100: Select P33 interrupt to automatically trigger ADC conversion
- 11101: Select P34 interrupt to automatically trigger ADC conversion
- 11110: Select P35 interrupt to automatically trigger ADC conversion
- 11111: Select P36 interrupt to automatically trigger ADC conversion

Note:

- 1) The TIM4/5/6 interrupt triggers the automatic conversion of ADC. In addition to enabling the corresponding interrupt of TIM4/5/6, it is also Need to configure Advanced Timer's spread spectrum and interrupt trigger selection register TIMX_CR selection can be triggered ADC interrupt source.
- 2) The rising edge of each interrupt flag bit is used to trigger the ADC. If you need to trigger repeatedly, you need to clear Off sign. If you do not need to enter the interrupt service routine, please do not enable the interrupt enable of the NVIC.

Page 475**21.10.3 ADC Configuration Register 2 (ADC_CR2)**

Offset address 0x00C

Reset value 0x00000000

31	30	29	28	27	26	25	twenty four	twenty five	twenty six	twenty seven	twenty eight	29	19	18	17	16
Reserved																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
			ADCCNT				CH7EN	CH6EN	CH5EN	CH4EN	CH3EN	CH2EN	CH1EN	CH0EN		

R/W R/W/R/W/R/W/R/W/R/W/R/W/R/W

Bit	mark	Function description
31:16	Reserved	Reserve
		ADC continuous conversion times configuration
		0: Continuous conversion 1 time
15:8	ADCCNT	1: Continuous conversion 2 times ... 255: 256 consecutive conversions
		ADC continuous conversion channel 7 enable
7	CH7EN	1: enable 0: prohibited
		ADC continuous conversion channel 6 enable
6	CH6EN	1: enable 0: prohibited
		ADC continuous conversion channel 5 enable
5	CH5EN	1: enable 0: prohibited
		ADC continuous conversion channel 4 enable
4	CH4EN	1: enable 0: prohibited
		ADC continuous conversion channel 3 enable
3	CH3EN	1: enable 0: prohibited
		ADC continuous conversion channel 2 enable
2	CH2EN	1: enable 0: prohibited
		ADC continuous conversion channel 1 enable
1	CH1EN	1: enable 0: prohibited
0	CH0EN	ADC continuous conversion channel 0 enable

Page 476

0: prohibited

Page 477

21.10.4 ADC channel 0 conversion result (**ADC_result0**)

Offset address 0x030

Reset value 0x00000000

31	30	29	28	27	26	25	twenty fourty thwenty twenty oné	19	18	17	16
Reserved											
15	14	13	12	11	10	9	8	7	6	5	4
3	2	1	0								

Reserved	result0 RO
----------	---------------

Bit	mark	Function description
31:12	Reserved	Reserve
11:0	result0	ADC channel 0 conversion result

21.10.5 ADC channel 1 conversion result (ADC_result1)

Offset address 0x034

Reset value 0x00000000

31 30 29 28 27 26 25 twenty fourty thventy twentyy on@0 19 18 17 16	Reserved														
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	result1 RO														
Reserved															

Bit	mark	Function description
31:12	Reserved	Reserve
11:0	result1	ADC channel 1 conversion result

21.10.6 ADC channel 2 conversion result (ADC_result2)

Offset address 0x038

Reset value 0x00000000

31 30 29 28 27 26 25 twenty fourty thventy twentyy on@0 19 18 17 16	Reserved														
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	result2 RO														
Reserved															

Bit	mark	Function description
31:12	Reserved	Reserve
11:0	result2	ADC channel 2 conversion result

21.10.7 ADC channel 3 conversion result (ADC_result3)

Offset address 0x03C

Reset value 0x00000000

31	30	29	28	27	26	25	twent	twent	twent	twent	on	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
result3															
RO															
Bit	mark														
31:12	Reserved														
11:0	result3														

Page 479**21.10.8 ADC channel 4 conversion result (ADC_result4)**

Offset address 0x040

Reset value 0x00000000

31	30	29	28	27	26	25	twent	twent	twent	twent	on	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
result4															
RO															
Bit	mark														
31:12	Reserved														
11:0	result4														

21.10.9 ADC channel 5 conversion result (ADC_result5)

Offset address 0x044

Reset value 0x00000000

31	30	29	28	27	26	25	twent	twent	twent	twent	on	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
result5															
RO															

Bit	mark	Function description
31:12	Reserved	Reserve
11:0	result5	ADC channel 5 conversion result

Page 480**21.10.10 ADC channel 6 conversion result (ADC_result6)**

Offset address 0x048

Reset value 0x00000000

31	30	29	28	27	26	25	twenty	fourty	thirty	twenty	twenty	one	19	18	17	16
Reserved																

b15 b14 b13 b12 b11 b10 b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
result6									
RO									

Bit	mark	Function description
31:12	Reserved	Reserve
11:0	result6	ADC channel 6 conversion result

21.10.11 ADC channel 7 conversion result (ADC_result7)

Offset address 0x04C

Reset value 0x00000000

31	30	29	28	27	26	25	twenty	fourty	thirty	twenty	twenty	one	19	18	17	16
Reserved																

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
result7															
RO															

Bit	mark	Function description
31:12	Reserved	Reserve
11:0	result7	ADC channel 7 conversion result

Page 481**21.10.12 ADC channel 8 conversion result (ADC_result8)**

Offset address 0x050

Reset value 0x00000000

31	30	29	28	27	26	25	twent	four	twent	twent	twent	on	0	19	18	17	16	
Reserved																		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
result8																		
RO																		

Bit mark Function description

31:12 Reserved Reserve

11:0 result8 ADC channel 8 conversion result

21.10.13 ADC conversion result accumulated value (ADC_result_acc)

Offset address 0x054

Reset value 0x00000000

31	30	29	28	27	26	25	twent	four	twent	twent	twent	on	0	19	18	17	16	
Reserved																		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
Result_acc[15:0]																		
RO																		

Bit mark Function description

31:20 Reserved Reserve

19:0 Result_acc ADC conversion accumulated value

Page 482**21.10.14 ADC comparison upper threshold (ADC廖HT)**

Offset address 0x058

Reset value 0x00000FFF

31	30	29	28	27	26	25	twenty	fourty	thirty	twenty	twenty	on off	19	18	17	16
Reserved																

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HT															
R/W															

Bit	mark	Function description
31:12	Reserved	Reserve
11:0	HT	ADC conversion result comparison upper threshold

21.10.15 ADC comparison lower threshold (ADC廖LT)

Offset address 0x05C

Reset value 0x00000000

31	30	29	28	27	26	25	twenty	fourty	thirty	twenty	twenty	on off	19	18	17	16
Reserved																

b15 b14 b13 b12 b11 b10 b9	b8	b7	b6	b5	b4	b3	b2	b1	b0				
LT													
R/W													

Bit	mark	Function description
31:12	Reserved	Reserve
11:0	LT	ADC conversion result comparison lower threshold

Page 483

21.10.16 ADC Interrupt Flag Register (ADC_IFR)

Offset address 0x060

Reset value 0x00000000

Bit	mark	Function description
31:4	Reserved	Reserve Continuous conversion complete flag
3	CONT_INTF	1: ADC continuous conversion completed 0: ADC continuous conversion is not completed ADC conversion result comparison interval flag
2	REG_INTF	1: ADC conversion result is in the range of [ADC_LT, ADC_HT) 0: ADC conversion result is outside the range of [ADC_LT, ADC_HT) ADC conversion result comparison upper threshold flag
1	HHT_INTF	1: ADC conversion result is in the interval of [ADC_HT, 4095] 0: ADC conversion result is outside the range of [ADC_HT, 4095] ADC conversion result comparison lower threshold flag
0	LLT_INTF	1: ADC conversion result is in the interval [0, ADC_LT) 0: ADC conversion result is outside the range of [0, ADC_LT)

21.10.17 ADC Interrupt Clear Register (ADC_ICLR)

Offset address 0x064

Reset value 0x00000004

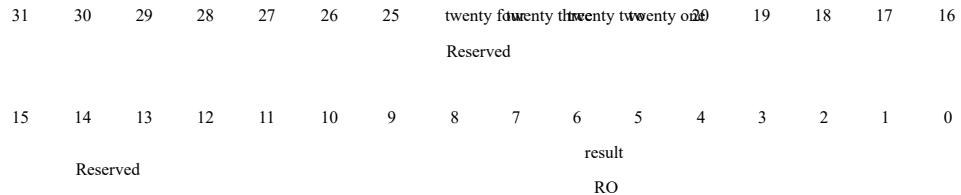


Bit	mark	Function description
31:4	Reserved	Reserve
3	CONT_INTC	Write 0 to clear the continuous conversion complete flag Write 1 has no effect
2	REG_INTC	Write 0 to clear the ADC conversion result comparison interval flag Write 1 has no effect
1	HHT_INTC	Write 0 to clear the upper threshold of ADC conversion result comparison Write 1 has no effect
0	LLT_INTC	Write 0 to clear the lower threshold flag of ADC conversion result comparison Write 1 has no effect

21.10.18 ADC result (ADC_result)

Offset address 0x068

Reset value 0x00000000



Bit	mark	Function description
31:12	Reserved	Reserve
11:0	result	ADC conversion result

22 analog comparator (VC)

22.1 Introduction to analog voltage comparator VC

The analog voltage comparator VC is used to compare the magnitude of the two input analog voltages and output high/low according to the comparison level. When the voltage at the "+" input terminal is higher than the voltage at the "-" input terminal, the output of the voltage comparator is high; when the voltage at the "+" input terminal is lower than the voltage at the "-" input terminal, the output of the voltage comparator is low. Inside this product, the integrated analog voltage comparator VC has the following characteristics:

Support voltage comparison function;

Supports internal 64-step VCC voltage division (the voltage of the divided voltage source needs to be greater than 1.8V);

Support 8 external input ports and the reference voltage output by the on-chip BGR as the input of the voltage comparator;

Support three software-configurable interrupt trigger modes: high level trigger/rising edge trigger/falling edge trigger;

The output of the voltage comparator can be used as the input of the Base Timer and LPTimer gate control ports;
The output of the voltage comparator can be used as the brake input or capture input of Advanced Timer;

Support work in ultra-low power mode, the interrupt output of the voltage comparator can change the chip from ultra-low power mode wake;

Provide software configurable filter time to enhance the anti-interference ability of the chip.

22.2 Frame Diagram of Voltage Comparator

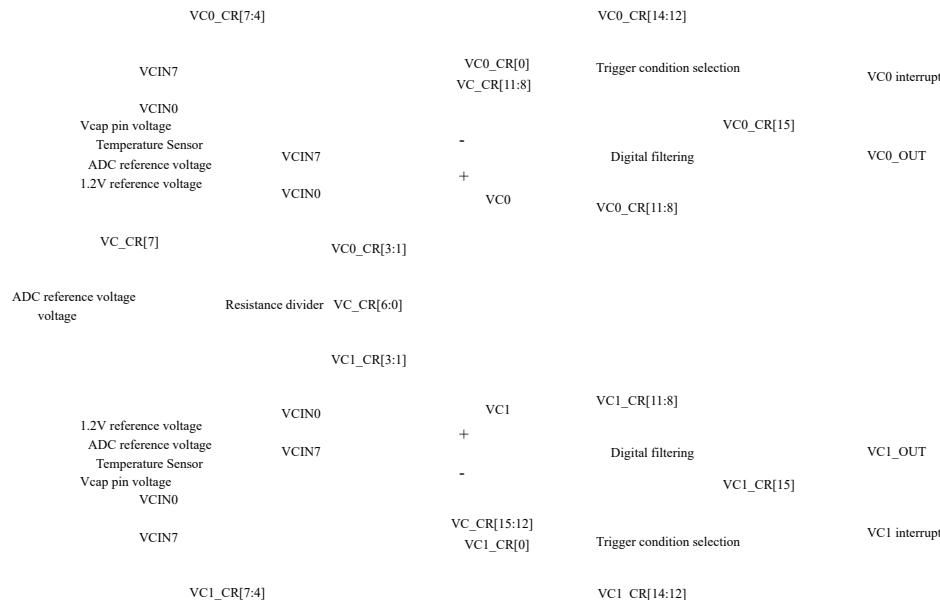


Figure 22-1 VC frame diagram

22.3 build / Response Time

When using a voltage comparator, from VC enable or the input voltage at both ends of VC changes to the output correct result

The time is determined by the BIAS_SEL control bit in the VC control register (VC_CR).

If you select temperature sensor, 1.2V reference voltage, ADC module reference voltage as the terminal input of the comparator, you need to

To open the internal BGR module. The start-up time of the internal BGR is about 20us, and the voltage comparator needs to wait

The output can be normal after the internal BGR is stable.

22.4 Filter time

In addition to the inherent setup/response time of the voltage comparator, the user can set a longer filter time to filter out the system

Noise, such as high current noise when the motor is stopped.

The signal level after digital filtering can be read from the register VC_IFR.VCx_Filter; when the GPIO function is configured as

When VCx_OUT, the digitally filtered signal can be output from GPIO to facilitate measurement.

Comparators
Output waveform

Digital filter circuit
Output waveform

Figure 22-2 VC filter response time

22.5 Hysteresis function

The voltage comparator can choose the hysteresis function, the figure after the hysteresis function is enabled is as follows

Negative terminal voltage + hysteresis

Negative terminal voltage-hysteresis

Comparator output waveform

Figure 22-3 VC hysteresis function

Page 488

22.6 VC Register

Base address 0x40002400

register	Offset address	describe
VC_CR	0x010	VC0/1 configuration register 0
VC0_CR	0x014	VC0 configuration register
VC1_CR	0x018	VC1 configuration register
VC0_OUT_CFG	0x01C	VC0 output configuration register
VC1_OUT_CFG	0x020	VC1 output configuration register
VC_IFR	0x024	VC interrupt register

Table 22-1 VC Register

Page 489**22.6.1 VC Configuration Register (VC_CR)**

Offset address 0x010

Reset value 0x00000020

31	30	29	28	27	26	25 24	twenty three twenty two twenty one	19	18	17	16
Reserved											
15	14	13	12	11	10	9	8	7	6	5	4
VC1_HYS _SEL	VC1_BIAS _SEL	VC0_HYS _SEL	VC0_BIA _SEL	VC_REF _SEL	VC_DIV _SEL	VC0_REF _SEL	VC0_EN	VC0_SEL _SEL	VC0_EN	VC0_SEL _SEL	VC0_EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	mark	Function description
31:16	Reserved	Reserve
15:14	VC1_HYS_SEL	VC1 hysteresis selection: 00: no lag 01: The hysteresis voltage is about 10mV 10: Hysteresis voltage is about 20mV 11: Hysteresis voltage is about 30mV
13:12	VC1_BIAS_SEL	VC1 power consumption selection (the greater the power consumption, the faster the response speed) 00:300nA 01:1.2uA 10:10uA (need to supply voltage not less than 2.8V, need to manually turn on BGR) 11:20uA (need to supply voltage not less than 2.8V, need to manually turn on BGR) Note: BGR startup time is about 30us
11:9	VC0_HYS_SEL	VC0 hysteresis selection: 00: no lag 01: The hysteresis voltage is about 10mV 10: Hysteresis voltage is about 20mV 11: Hysteresis voltage is about 30mV
9:8	VC0_BIAS_SEL	VC0 power consumption selection (the greater the power consumption, the faster the response speed) 00:300nA 01:1.2uA 10:10uA (need to supply voltage not less than 2.8V, need to manually turn on BGR) 11:20uA (need to supply voltage not less than 2.8V, need to manually turn on BGR) Note: BGR startup time is about 30us
7	VC_REF2P5_SEL	VC_DIV reference voltage Vref selection 0: VCC 1: The reference voltage selected by ADC_CR0.SREF
6	VC_DIV_EN	6-bit DAC enable

Page 490

```

1: enable
6-bit DAC configuration
000000: 1/64 Vref
000001:2/64 Vref
000010:3/64 Vref
000011:4/64 Vref
...
111110:63/64 Vref
111111: Vref

```

22.6.2 VC0 configuration register (VC0_CR)

Offset address 0x014

Reset value 0x00000000

31	30	29	28	27	26	25	twenty four	22	twenty one	19	18	17	16
Reserved													

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE	level	rising	falling		debounce_time		FLTEN		n_sel		p_sel		EN		
R/W	WR	WR	WR/W		R/W		R/W		R/W		R/W		R/W		

Bit	mark	Function description
31:16	Reserved	Reserve
15	IE	VC interrupt enable 1: enable; 0: disable
14	level	VC output signal high level triggers INT flag
13	rising	The rising edge of the VC output signal triggers the INT flag
12	Falling	VC output signal falling edge triggers INT flag
		VC output filter time configuration 111: The filtering time is about 28.8ms 110: The filtering time is about 7.2ms 101: The filtering time is about 1.8ms 100: The filtering time is about 450us 011: The filtering time is about 112us 010: The filtering time is about 28us 001: The filtering time is about 14us 000: The filtering time is about 7us Note: The filter time configuration is only valid when FLTEN=1.
8	FLTEN	1: Start VC filtering 0: VC without filtering Voltage comparator "-" terminal input selection 0000: select channel 0 input P2.3 0001: select channel 1 input P2.5 0010: select channel 2 input P3.2 0011: select channel 3 input P3.3 0100: select channel 4 input P3.4 0101: select channel 5 input P3.5 0110: select channel 6 input P3.6 0111: select channel 7 input P0.1 1000: Resistance divider output voltage
7:4	N_SEL	
3:1	P_SEL	
0	EN	

1001: Built-in temperature sensor output voltage
1010: internal reference 1.2V output voltage
1011: Reference voltage of ADC module
1100: Voltage of the VCAP pin
Voltage comparator "+" terminal input selection 000: select channel 0 input P2.3 001: select channel 1 input P2.5 010: select channel 2 input P3.2 011: select channel 3 input P3.3 100: select channel 4 input P3.4 101: select channel 5 input P3.5 110: select channel 6 input P3.6 111: select channel 7 input P0.1 Voltage comparator enable 1: Enable voltage comparator 0: Turn off the voltage comparator

Page 493**22.6.3 VC1 Configuration Register (VC1_CR)**

Offset address 0x018

Reset value 0x00000000

31	30	29	28	27	26 25	twenty four 22	twenty one 20	19	18	17	16
Reserved											
15	14	13	12	11	10	9	8	7	6	5	4
IE	level	rising	falling	debounce_time	FLTEN	n_sel	p_sel				EN

Bit	mark	Function description
31:16	Reserved	Reserve
15	IE	VC interrupt enable 1: enable; 0: disable
14	level	VC output signal trigger interrupt selection 1: Enable high level trigger INT flag 0: Prohibit high level triggering INT flag
13	rising	VC output signal trigger interrupt selection 1: Enable rising edge to trigger INT flag 0: Prohibit rising edge to trigger INT flag
12	falling	VC output signal trigger interrupt selection 1: Enable falling edge to trigger INT flag

0: Prohibit falling edge to trigger INT flag
 VC output filter time configuration
 111: The filtering time is about 28.8ms
 110: The filtering time is about 7.2ms
 101: The filtering time is about 1.8ms
 100: The filtering time is about 450us
 011: The filtering time is about 112us
 010: The filtering time is about 28us
 001: The filtering time is about 14us
 000: The filtering time is about 7us
 Note: The filter time configuration is only valid when FLTEN=1.

8 FLTEN 1: Start VC filtering
 0: VC without filtering
 Voltage comparator "-" terminal input selection
 7:4 N_SEL 0000: select channel 0 input P2.3
 0001: select channel 1 input P2.5
 0010: select channel 2 input P3.2

Page 494

0011: select channel 3 input P3.3
 0100: select channel 4 input P3.4
 0101: select channel 5 input P3.5
 0110: select channel 6 input P3.6
 0111: select channel 7 input P0.1
 1000: Resistance divider output voltage
 1001: Built-in temperature sensor output voltage
 1010: internal reference 1.2V output voltage
 1011: Reference voltage of ADC module
 1100: Voltage of the VCAP pin
 Voltage comparator "+" terminal input selection
 000: select channel 0 input P2.3
 001: select channel 1 input P2.5
 010: select channel 2 input P3.2
 3:1 P_SEL 011: select channel 3 input P3.3
 100: select channel 4 input P3.4
 101: select channel 5 input P3.5
 110: select channel 6 input P3.6
 111: select channel 7 input P0.1
 Voltage comparator enable
 0 EN 1: Enable voltage comparator
 0: Turn off the voltage comparator

Page 495

22.6.4 VC0 output configuration register (VC0_OUT_CFG)

Offset address 0x01C

Reset value 0x00000000

31	30	29	28	27	26	25	twenty four	twenty three	twenty twenty one	19	18	17	16
Reserved													
15	14	13	12	11	10	9	8	7	6	5	4	3	2
		INV_T		INV_T		INV_T	Pcae	PCAC	INV_-	TM3E	LPTI	TIM2	TIM1
brake	TIM6		TIM5		TIM4		CI	AP0	PCA	CLK	MG	G	G
		imer6		imer5		imer4							Timer

Bit	mark	Function description
31:16	Reserved	Reserve
15	brake	VC0 as Advanced Timer brake control 1: Enable; 0: Disable.
14	TIM6	VC0 filter result output to TIM6 capture input enable 1: Enable; 0: Disable.
13	INV_TIM6	VC0 filter result output to TIM6 reverse enable 1: Enable reverse direction; 0: Disable reverse direction, input and VC output are in the same direction.
12	TIM5	VC0 filter result output to TIM5 capture input enable 1: Enable; 0: Disable.
11	INV_TIM5	VC0 filter result output to TIM5 reverse enable 1: Enable reverse direction; 0: Disable reverse direction, input and VC output are in the same direction.
10	TIM4	VC0 filter result output to TIM4 capture input enable 1: Enable; 0: Disable.
9	INV_TIM4	VC0 filter result output to TIM4 reverse enable 1: Enable reverse direction; 0: Disable reverse direction, input and VC output are in the same direction.
8	PCAECl	VC0 filter result is output to PCA external clock enable control 1: Enable; 0: Disable.
7	PCACAP0	VC0 filter result output to PCA capture 0 enable control 1: Enable; 0: Disable.
6	INV_PCA	VC0 filter result output negative to PCA 1: Enable reverse direction; 0: Disable reverse direction, input and VC output are in the same direction.
5	LPTIMECLK	VC0 filter result is output to LPTIMER external clock enable control 1: Enable; 0: Disable.
4	LPTIMG	VC0 filter result is output to LPTIMER3 GATE enable control 1: Enable; 0: Disable.
3	TIM2G	VC0 filter result is output to TIM2 GATE enable control

1: Enable; 0: Disable.

Page 496

2	TIM1G	VC0 filter result is output to TIM1 GATE enable control 1: Enable; 0: Disable.
1	TIM0G	VC0 filter result is output to TIM0 GATE enable control 1: Enable; 0: Disable.
0	INV_Timer	VC0 filter result output negative to each TIM0/1/2, LPTimer 1: Enable reverse direction; 0: Disable reverse direction, input and VC output are in the same direction.

Page 497

22.6.5 VC1 output configuration register (VC1_OUT_CFG)

Offset address 0x020

Reset value 0x00000000

Bit	mark	Function description
31:16	Reserved	Reserve
15	brake	VC1 as Advanced Timer brake control 1: Enable; 0: Disable.
14	TIM6	VC1 filter result output to TIM6 capture input enable 1: Enable; 0: Disable.
13	INV_TIM6	VC1 filter result output to TIM6 reverse enable 1: Enable reverse direction; 0: Disable reverse direction, input and VC output are in the same direction.
12	TIM5	VC1 filter result output to TIM5 capture input enable 1: Enable; 0: Disable.
11	INV_TIM5	VC1 filter result output to TIM5 reverse enable 1: Enable reverse direction; 0: Disable reverse direction, input and VC output are in the same direction.
10	TIM4	VC1 filter result output to TIM4 capture input enable 1: Enable; 0: Disable.
9	INV_TIM4	VC1 filter result output to TIM4 reverse enable 1: Enable reverse direction; 0: Disable reverse direction, input and VC output are in the same direction.
8	PCAECI	VC1 filter result is output to PCA external clock enable control 1: Enable; 0: Disable.
7	PCACAP1	VC1 filter result output to PCA capture 1 enable control 1: Enable; 0: Disable.
6	INV_PCA	VC1 filter result output negative to PCA 1: Enable reverse direction; 0: Disable reverse direction, input and VC output are in the same direction.
5	LPTIMECLK	VC1 filter result is output to LPTIMER external clock enable control 1: Enable; 0: Disable.
4	LPTIMG	VC1 filter result is output to LPTIMER3 GATE enable control 1: Enable; 0: Disable.
3	TIM2G	VC1 filter result is output to TIM2 GATE enable control 1: Enable; 0: Disable.

2	TIM1G	VC1 filter result is output to TIM1 GATE enable control 1: Enable; 0: Disable.
1	TIM0G	VC1 filter result is output to TIM0 GATE enable control 1: Enable; 0: Disable.
0	INV_Timer	VC1 filter result output negative to each TIM0/1/2, LPTimer 1: Enable reverse direction; 0: Disable reverse direction, input and VC output are in the same direction.

Page 499**22.6.6 VC Interrupt Register (VC_IFR)**

Offset address 0x024

Reset value 0x00000000

31	30	29	28	27	26	25	twenty four	twenty three	twenty two	twenty one	19	18	17	16	
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															
VC1_ VC0_ VC1_ VC0_ Filter Filter INTF INTF RO RO R/W/R/W															

Bit	mark	Function description
31:4	Reserved	Reserve

3	VC1_Filter	Status after VC1 Filter
2	VC0_Filter	Status after VC0 Filter
1	VC1_INTF	VC1 interrupt flag, 1 VC1 interrupt occurs; 0 does not occur interrupt; write 0 to clear the interrupt flag, write 1 invalid
0	VC0_INTF	VC0 interrupt flag, 1 VC0 interrupt occurs; 0 does not occur interrupt; write 0 to clear the interrupt flag, write 1 invalid

23 Low Voltage Detector (LVD)

23.1 Introduction to LVD

LVD can be used to monitor the voltage of VCC and chip pins. When the comparison result of the monitored voltage and the LVD threshold is satisfied

When the condition is triggered, the LVD will generate an interrupt or reset signal, and the user can perform some urgent tasks based on this signal.

LVD has the following characteristics:

4 monitoring sources, VCC, P03, P23, P25;

16-level threshold voltage, flexible and versatile;

8 trigger conditions, high level, rising edge, falling edge combination;

2 trigger results, reset and interrupt;

8 order filter configuration to prevent false triggering;

With hysteresis function, strong anti-interference.

23.2 LVD block diagram

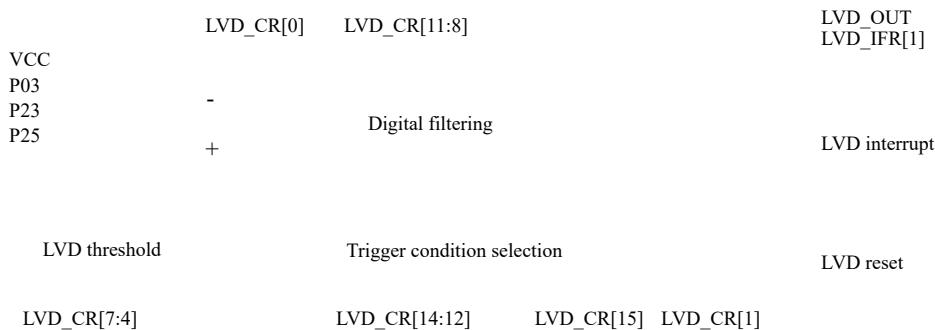


Figure 23-1 LVD block diagram

23.3 Digital filtering

If the working environment of the chip is bad, the output of the hysteresis comparator will have a noise signal. Enable the digital filter module, In the output waveform of the hysteresis comparator, the noise signal whose pulse width is less than LVD_CR.Debounce_time can be filtered. remove. If the digital filter module is disabled, the input and output signals of the digital filter module are the same. The signal level after digital filtering can be read from the register LVD_IFR[1]; when the GPIO function is configured as LVD_OUT At the time, the digitally filtered signal can be output from GPIO to facilitate measurement.

Enable the digital filter module, the filter diagram is as follows:

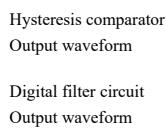


Figure 23-2 LVD filter output

23.4 Hysteresis function

The built-in voltage comparator of LVD has a hysteresis function, and its output signal will wait until the input signal is higher or lower than the threshold It turns over after 20mV. The hysteresis function can enhance the anti-interference ability of the chip, as shown in the figure below.

Threshold voltage +20mV

Threshold voltage -20mV

Hysteresis comparator
Output waveform

Figure 23-3 LVD hysteresis response

Page 502

23.5 configuration example

23.5.1 LVD is configured as low voltage reset

In this mode, the MCU is reset when the monitoring voltage is lower than the threshold voltage.

The configuration method is as follows:

- Step1: Configure LVD_CR.Source_sel, select the voltage source to be monitored.
- Step2: Configure LVD_CR.VTDS and select the threshold voltage of LVD.
- Step3: Configure LVD_CR.Debounce_time and select LVD filter time.
- Step4: Configure LVD_CR.FLTEN and enable LVD filtering.
- Step5: Set LVD_CR.HTEN to 1, select high level to trigger LVD action.
- Step6: Set LVD_CR.ACT to 1, and select LVD action as reset.
- Step7: Set LVD_CR.LVDEN to 1, enable LVD.

23.5.2 LVD is configured as voltage change interrupt

In this mode, an interrupt is generated when the monitoring voltage is higher or lower than the threshold voltage.

The configuration method is as follows:

- Step1: Configure LVD_CR.Source_sel, select the voltage source to be monitored.
- Step2: Configure LVD_CR.VTDS and select the threshold voltage of LVD.
- Step3: Configure LVD_CR.Debounce_time and select LVD filter time.
- Step4: Configure LVD_CR.FLTEN and enable LVD filtering.
- Step5: Set LVD_CR.RTEN and LVD_CR.FTEN to 1, and select level changes to trigger LVD action.
- Step6: Set LVD_CR.ACT to 0 and select LVD action as interrupt.
- Step7: Set LVD_CR.IE to 1, enable LVD interrupt.
- Step8: Enable the LVD interrupt in the NVIC interrupt vector table.
- Step9: Set LVD_CR.LVDEN to 1, enable LVD.
- Step10: Write 0x00 to LVD_IFR in the interrupt service routine to clear the interrupt flag.

Page 503**23.6 LVD Register**

Base address 0x40002400

register	Offset address	describe
LVD_CR	0x028	LVD configuration register
LVD_IFR	0x02C	LVD interrupt flag register

Table 23-1 LVD Register

23.6.1 LVD Configuration Register (LVD_CR)

Offset address 0x028

Reset value 0x00000100

31	30	29	28	27	26	25	twenty four	twent	twent	twent	on	19	18	17	16										
Reserved																									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0										
IE	HTEN	RTEN	FTEN	Debounce_time			FLT EN	VTDS			Source_sel		ACT EN	LVD											
R/W	R/W	R/W	R/W	R/W			R/W	R/W			R/W		R/WR/W												
Bit	mark	Function description																							
31:16	Reserved	Reserve																							
15	IE	LVD interrupt enable																							
14	HTEN	1: enable; 0: Prohibited. High level trigger enable (the monitored voltage is lower than the threshold voltage)																							
13	RTEN	1: enable; 0: Prohibited. Rising edge trigger enable (the monitored voltage changes from higher than the threshold voltage to lower than the threshold voltage)																							
12	FTEN	1: enable; 0: Prohibited. Digital filter time configuration 111: The filtering time is about 28.8ms 110: The filtering time is about 7.2ms 101: The filtering time is about 1.8ms 100: The filtering time is about 450us																							

Page 504

011: The filtering time is about 112us
 010: The filtering time is about 28us
 001: The filtering time is about 14us
 000: The filtering time is about 7us
 Note: The filter time is only valid when FLTEN is 1.

Digital filtering function configuration

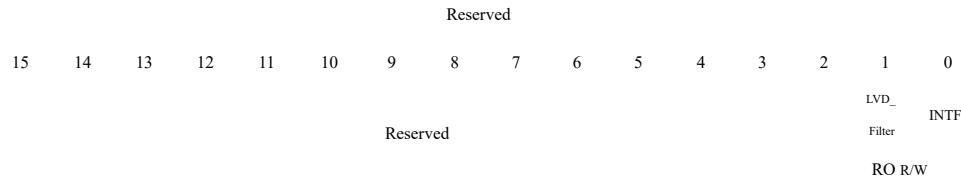
8	FLTEN	1: Enable digital filter 0: Disable digital filtering
		LVD monitoring voltage selection
		1111: 3.3v
		1110: 3.2v
		1101: 3.1v
		1100: 3.0v
		1011: 2.9v
		1010: 2.8v
		1001: 2.7v
7:4	VTDS	1000: 2.6v 0111: 2.5v 0110: 2.4v 0101: 2.3v 0100: 2.2v 0011: 2.1v 0010: 2.0v 0001: 1.9v 0000: 1.8v
		LVD monitoring source selection
		11: P2.5 port input voltage
3:2	Source_sel	10: P2.3 port input voltage 01: P0.3 port input voltage 00: VCC power supply voltage
		LVD trigger action selection
1	ACT	1: System reset 0: NVIC interrupt
		LVD control
0	LVDEN	1: Enable LVD 0: Disable LVD

23.6.2 LVD Interrupt Register (LVD_IFR)

Offset address 0x02C

Reset value 0x00000000

31	30	29	28	27	26	25	twenty fourty twenty twenty twenty ond	19	18	17	16
----	----	----	----	----	----	----	--	----	----	----	----



24 simulate other registers

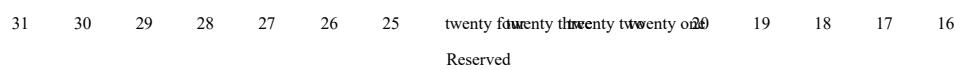
Base address 0x40002400

register	Offset address	describe
BGR_option	0x078	BGR control register

24.1 BGR Configuration Register (BGR_CR)

Offset address 0x000

Reset value 0x00000000



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														TS_	BGR
														EN	_EN

Reserved

R/W/R/W

Bit	mark	Function description
31:2	Reserved	Reserve Internal temperature sensor enable
1	TS_EN	1: Start the internal temperature sensor 0: Disable the internal temperature sensor Note: TS needs about 20us startup time to stabilize. BGR enable control 1: Enable BGR 0: Disable BGR Notice: 0 BGR_EN 1) This register can be operated only when PERI_CLKEN.ADC is 1. 2) The stable and high-precision reference voltage can be output only after the BGR is enabled for 20us. After BGR is stable, it can be used by other The module is used, so the step of waiting for the BGR to stabilize should be added to the user's operation. 3) When using ADC, BGR must be enabled. 4) When using VC, it is necessary to decide whether to enable BGR according to the configuration of the VC register.
0		

25 SWD debug interface

This series uses the ARM Cortex-M0+ core, which has a hardware debugging module SWD, which supports complex debugging operate. The hardware debug module allows the kernel to stop when fetching instructions (instruction breakpoints) or accessing data (data breakpoints) At the end, both the internal state of the kernel and the external state of the system can be queried in the IDE. After completing the query, within The core and peripherals can be restored, and the program will continue to execute. When the HC32L110 microcontroller is connected to the debugger When debugging, the debugger will use the kernel's hardware debugging module for debugging operations.

Notice:

- SWD cannot work in DeepSleep mode. Please perform debugging in Active and Sleep mode.

25.1 SWD debugging additional functions

This product uses the ARM Cortex-M0+ CPU, the core contains hardware extensions for advanced debugging functions, because The debugging function of this product is consistent with Cortex-M0+. The debug extension allows the kernel to fetch instructions (instructions Stop the kernel when accessing data (data breakpoint) or access data (data breakpoint). When the kernel is stopped, you can query the internal state of t State and the external state of the system. After the query is completed, the kernel and system will be restored and program execution will resume. When the debugging host is connected to the MCU and debugging, the debugging function will be used.

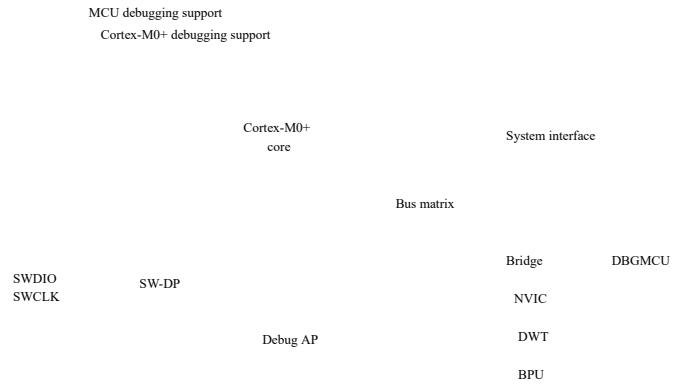


Figure 25-1 Block diagram of debugging support

Page 508

The debug function built into the Cortex®-M0+ core is part of the ARM® CoreSight design suite.

The ARM® Cortex®-M0+ core provides integrated on-chip debugging support. It includes:

SW-DP: serial line

BPU: Breakpoint unit

DWT: data observation point trigger

Note:

- For detailed information about the debugging functions supported by the ARM® Cortex®-M0+ core, see Cortex®-M0+ Technical Reference Manual.

25.2 ARM® Reference Document

Cortex®-M0+ Technical Reference Manual (TRM)

Available from www.infocenter.arm.com.

ARM® debug interface V5

ARM® CoreSight Design Suite Version r1p1 Technical Reference Manual

Page 509

25.3 Debug Port Pins

25.3.1 SWD port pins

The SWD interface of this series requires 2 pins, as shown in the following table.

SWD port name	Debug function	Pin assignment
SWCLK	Serial clock	P31
SWDIO	Serial data input/output	P27

25.3.2 SW-DP pin assignment

If the [Encryption Chip] option is enabled when programming the program, the SWD debugging function will be disabled after power-on. If burn

If the [Encryption Chip] option is not enabled in the program, the P27/P31 pins will be initialized to be used by the debugger after power-on

Dedicated pins used. The user can set the SYSCTRL1.SWD_USE_IO register to disable the SWD pin

For debugging function, the SWD pin will be released to be used as a normal GPIO. The configuration and function summary of the SWD pin is as fol

Shown:

[Encryption chip] option	SWD_USE_IO configuration	P27/P31 function
encryption	0	NA
encryption	1	GPIO
Not encrypted	0	SWD
Not encrypted	1	GPIO

25.3.3 Internal pull-up on the SWD pin

After the user software releases the SW I/O, the GPIO controller will control these pins. GPIO control register reset

The state puts the I/O in an equivalent state:

SWDIO: Input pull up

SWCLK: Input pull-up

Because of the built-in pull-up and pull-down resistors, there is no need to add external resistors.

25.4 SWD port

25.4.1 Introduction to SWD Protocol

This synchronous serial protocol uses two pins:

SWCLK: the clock from the host to the target

SWDIO: Two-way

Using this protocol, two sets of register sets (DPACC register set and APACC register set) can be read and written at the same time (Device group). When transmitting data, LSB comes first.

For SWDIO bidirectional management, the line must be pulled up on the circuit board (ARM® recommends 100 k).

These pull-up resistors can be configured internally. No external pull-up resistor is required.

Every time the direction of SWDIO is changed in the protocol, the conversion time will be inserted, and the line will not be driven by the host at this time. Nor is it driven by goals. By default, this conversion time is one bit time, but you can configure the SWCLK frequency Rate to adjust.

25.4.2 SWD protocol sequence

Each sequence includes three stages:

1. Data packet request sent by the host (8 bits)
2. Confirmation response sent by the target (3 bits)
3. The data transmission stage sent by the host or target (33 bits)

Bit	name	illustrate
0	start up	Must be 1
1	APnDP	0: DP access; 1: AP access
2	RnW	0: write request; 1: read request
4:3	A[3:2]	Address field of DP or AP register
5	Parity check	Unit parity of the first few bits
6	stop	0
7	Reside	Not driven by the host. Because of the pull-up, it must be read as 1 by the target

For a detailed description of the DPACC and APACC registers, please refer to Cortex®-M0+ TRM.

The packet request is always followed by the conversion time (default is 1 bit), at this time neither the host nor the target will be driven.

Bit	name	illustrate
0	ACK	001: FAULT 010: WAIT 100: OK
		Only when a READ transaction occurs or a WAIT or FAULT acknowledgement is received, the ACK response must be conversion time.
		Conversion time.

Bit	name	illustrate
0:31	WDATA or RDATA	Write or read data
32	Parity check	Single parity of 32 data bits

Only when a READ transaction occurs, it must be the conversion time after the DATA transfer.

25.4.3 SW-DP state machine (reset, idle state, ID code)

The state machine of SW-DP has an internal ID code for identifying SW-DP. This code complies with JEP-106 standard allow. This ID code is the default ARM® code and is set to **0x0BB11477** (equivalent to Cortex®-M0+).

Notice:

- Before the target reads this ID code, the SW-DP state machine is not working.

After power-on reset or the line is at a high level for more than 50 cycles, the SW-DP state machine is in the reset state state.

If the line is low for at least two cycles after the reset state, the SW-DP state machine is in the idle state.

After the reset state, the state machine must first enter the idle state, and then check the DP-SW ID CODE register

Perform read access. Otherwise, the target will issue a FAULT confirmation response on another transaction.

For more detailed information about the SW-DP state machine, please refer to *Cortex®-M0+ TRM* and *CoreSight Design Kit rIp0TRM*.

25.4.4 DP and AP read / write access

Does not delay the read access to DP: the target response can be sent immediately (if ACK=OK), or it can be delayed

Send the target response (if ACK=WAIT).

Delay the read access to AP. This means that the access result will be returned in the next transfer. If you want to execute the next

The second access is not an AP access, you must read the DP-RDBUFF register to get the result.

The DP-CTRL/STAT register will be updated every time AP read access or RDBUFF read request is made.

READOK flag to know whether the AP read access is successful.

SW-DP has a write buffer (for DP or AP write), so that even when other operations are still not completed,

Can also accept write operations. If the write buffer is full, the target confirms that the response is WAIT. But IDCODE

Except for read, CTRL/STAT read, or ABORT write, these operations will also occur when the write buffer is full been accepted.

Because there are asynchronous clock domains SWCLK and HCLK, after the write operation (after the parity bit), you need
Two additional SWCLK cycles to make the write operation take effect internally. Should drive the line low
These cycles are applied when (idle state).

This is especially important when writing to the CTRL/STAT register to make a power-on request. Otherwise the next operation (Operations that are valid only after the core is powered on) will be executed immediately, which will cause a failure.

25.4.5 SW-DP Register

These registers can be accessed when APnDP=0:

A[3:2]	RW	SELECT register The CTRLSEL bit	register	Annotation
00	Read		IDCODE	The manufacturer code is set to the default ARM ® of Cortex ® -M0+ Code. 0x0BB11477 (identification SW-DP)
00	Write		ABORT	Purpose: – Request system or debug power on
01	Read/write	0	DP- CTRL/STAT	– Configure the transmission operation of AP access – Control comparison and verification operations – Read some status flags (overflow and power-on confirmation)
01	Read/write	1	WIRE CONTROL	Used to configure the physical serial port protocol (such as the duration of the conversion time time)
10	Read		READ RESEND	Allows to recover the read data from the corrupted debugging software transmission, There is no need to repeat the original AP transmission.
10	Write		SELECT	4-word register window for selecting the current access port and activity
11	Read/write		READ BUFFER	Since AP access has been issued, this read buffer is very useful (in Provide the result of reading the AP request when executing the next AP transaction). This read buffer captures the data in the AP and is displayed as

The result of the previous reading, no need to start a new operation.

25.4.6 SW-AP Register

These registers can be accessed when APnDP=1:

There are multiple AP registers, which are addressed in the following combinations:

Shift value A[3:2]

The current value of the DP SELECT register

address	A[3:2] value	illustrate
0x0	00	Reserved, the reset value must be maintained.
0x4	01	DP CTRL/STAT register. Used for: – Request system or debug power on – Configure the transmission operation of AP access

		<ul style="list-style-type: none"> - Control comparison and verification operations -Read some status flags (overflow and power-on confirmation)
0x8	10	<p>DP SELECT register: used to select the current access port and active 4-word register window mouth.</p> <ul style="list-style-type: none"> - Bits 31:24: APSEL: select the current AP - Bit 23:8: reserved - Bit 7:4: APBANKSEL: Select the active 4-word register window on the current AP - Bits 3:0: reserved
0xC	11	<p>DP RDBUFF register: used to obtain the final result after performing a series of operations through the debugger fruit</p> <p>(No need to request a new JTAG-DP operation)</p>

25.5 Kernel debugging

Debug the kernel through the kernel debug register. Debug access to these registers through the debug access port. It is sent by four

Memory composition:

register	illustrate
DHCSR	32 -bit debug stop control and status register This register provides information about the state of the processor, enables the core to enter the debug stop state and provides the processor stepping function can.
DCRSR	17 -bit debug core register selector register: This register selects the processor register that needs to be read and written.
DCRDR	32 -bit debug core register data register: This register is stored between the register and the processor selected by the DCRSR (selector) register to read and write data.
DEMCR	32 -bit debug exception and monitoring control register: This register provides vector capture and debug monitoring control.

These registers are not reset when the system is reset. They can only be reset by power-on reset. For more details,

See Cortex®-M0+ TRM.

In order to make the kernel enter the debug stop state immediately after reset, you must:

Enable debug and exception monitoring control register bit 0 (VC_CORRESET)

Enable debug stop control and bit 0 of the status register (C_DEBUGEN)

25.6 BPU (Breakpoint Unit)

The Cortex®-M0+ BPU implementation provides four breakpoint registers.

25.6.1 BPU function

The processor breakpoint implements the PC-based breakpoint function.

For more information about the BPU CoreSight identification register and its address and access type, please refer to ARMv6-M ARM® and ARM® CoreSight component technical reference manual.

25.7 DWT (Data Observation Point)

The Cortex®-M0+ DWT implementation provides two watchpoint register sets.

25.7.1 DWT function

The processor observation point realizes the data address and the PC-based observation point function (that is, the PC sampling register), and supports Support the comparator address mask, as described in ARMv6-M ARM®.

25.7.2 DWT Program Counter Sampling Register

The processor that implements the data observation point unit also implements the ARMv6-M optional DWT program counter sampling register (DWT_PCSR). This register allows the debugger to sample the PC periodically without stopping the processor. This can provide coarse Slightly analyze. For more information, see ARMv6-M ARM®.

Cortex®-M0+ DWT_PCSR records the condition codes and instructions that pass and the instructions that fail the condition code.

Page 516

25.8 MCU debug component (DBG)

The MCU debugging component helps the debugger provide support for:

Low power consumption mode

Timer and watchdog clock control during breakpoint

25.8.1 Debugging support for low-power mode

To enter the low-power mode, the instruction WFI or WFE must be executed.

The MCU supports multiple low-power modes, which can disable the CPU clock or reduce CPU power consumption.

The kernel does not allow FCLK or HCLK to be turned off during a debugging session. Since they need to be used during debugging

Debug the connection, so it must remain active. The MCU integrates a special method that allows users to operate in low-power mode

Debug the software.

25.8.2 Debugging support for timers and watchdogs

During the breakpoint, the behavior of the timer and watchdog counter must be selected:

When a breakpoint is generated, the counter continues to count. For example, when PWM controls a motor, this method is usually required.

Mode.

When a breakpoint is generated, the counter stops counting. This method is required when used as a watchdog.

Page 517**25.9 Debug mode module working status control (DEBUG_ACTIVE)**

Reset value 0x00000FFF (Only in SWD debugging mode, this register setting is effective)

Offset address: 0x038

31	30	29	28	27	26	25	twenty	twenty	twenty	twenty	o20	19	18 17	16
Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
				LPTIM		RTC	WDT	PCA	TIM6	TIM5	TIM4	LPTIM	TIM2	TIM1
				Reserved		Res.	RW	RW	RW	RW	RW	RW	RW	RW

Bit	mark	Function description
31:12	Reserved	Reserve When debugging, Timer3 count function configuration
11	LPTIM	1: In the SWD debugging interface, pause Timer3 counting function 0: In the SWD debugging interface, Timer3 normal counting function
10	Reserved	Reserve When debugging, RTC counting function configuration
9	RTC	1: In the SWD debugging interface, pause the RTC counting function 0: In the SWD debugging interface, the RTC normal counting function When debugging, WDT counting function configuration
8	WDT	1: In the SWD debugging interface, pause the WDT counting function 0: In the SWD debugging interface, WDT normal counting function When debugging, PCA counting function configuration
7	PCA	1: In the SWD debugging interface, pause the PCA counting function 0: PCA normal counting function under SWD debugging interface When debugging, Timer6 counting function configuration
6	TIM6	1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debugging interface, Timer normal counting function When debugging, Timer5 counting function configuration
5	TIM5	1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debugging interface, Timer normal counting function When debugging, Timer4 counting function configuration
4	TIM4	1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debugging interface, Timer normal counting function When debugging, LpTimer counting function configuration
3	LPTIM	1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debugging interface, Timer normal counting function When debugging, Timer2 counting function configuration
2	TIM2	1: In the SWD debugging interface, pause the Timer counting function

Page 518

		0: In the SWD debugging interface, Timer normal counting function When debugging, Timer1 counting function configuration
1	TIM1	1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debugging interface, Timer normal counting function When debugging, Timer0 counting function configuration
0	TIM0	1: In the SWD debugging interface, pause the Timer counting function 0: In the SWD debugging interface, Timer normal counting function

26 device electronic signature

The electronic signature is stored in the system storage area of the flash memory module and can be read by SWD or CPU. it

The chip identification information included is written at the factory, and the user's firmware or external devices can read the electronic signature for

Automatically match HC32Fxxx / HC32Lxxx microcontrollers of different configurations.

26.1 Product unique identification (**UID**) register (**80** bits)

Typical application scenarios of unique identifiers:

Used as a serial number

It is used as security when using UID in combination with software encryption primitives and protocols before programming the internal Flash

Key to improve the security of the code in Flash

Activate the safety bootstrapping process, etc.

The 80-bit unique device identifier provides a reference number that is unique to any device and any context. use

The user can never change these bits. The 80-bit unique device identifier can also be in different ways such as single byte/half word/word, etc.

Read it, and then connect it using a custom algorithm.

Base address: 0x0010 0E74

Offset address	describe	UID Bit(80bit)
		7 6 5 4 3 2 1 0
0	X Coordinate on the wafer	UID[7:0]
1	Rev ID	UID[15:8]
2	Fixed value FFH	Reserved
3	Fixed value FFH	Reserved
4	Fixed value 00H	UID[23:16]
5	Fixed value 00H	UID[31:24]
6	Wafer Number	UID[39:32]
7	Y Coordinate on the wafer	UID[47:40]
8		UID[55:48]
9		UID[63:56]
10	Wafer Lot Number	UID[71:64]
11		UID[79:72]

26.2 Product Model Register

0x0010 0C60 ~ 0x0010 0C6F stores the ASCII code of the product model. If the product model is less than 16 bytes,

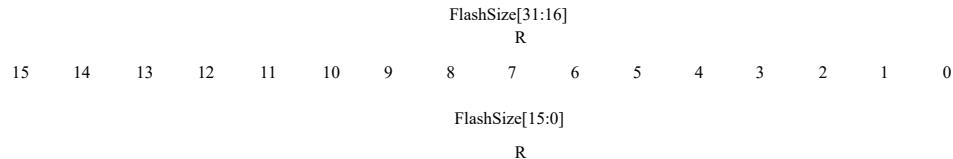
Fill it with 0x00.

Example: 484333324C3133364B38544100000000 represents the product model HC32L136K8TA.

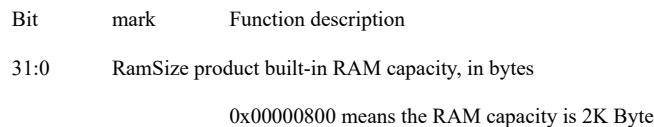
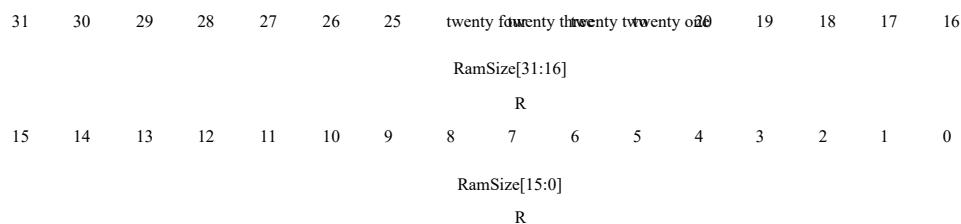
26.3 FLASH Capacity Register

Base address: 0x0010 0C70

31	30	29	28	27	26	25	twenty	twenty	twenty	twenty	twenty	0x00	19	18	17	16
----	----	----	----	----	----	----	--------	--------	--------	--------	--------	------	----	----	----	----

**Page 521****26.4 RAM Capacity Register**

Base address: 0x0010 0C74

**26.5 Number of pins register**

Base address: 0x0010 0C7A

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

PinCount[15:0]

R

Bit mark Function description

15:0 PinCount The number of product pins, in units

0x0020 represents the number of product pins is 32

27 Appendix A SysTick Timer

27.1 Introduction to SysTick Timer

If the OS wants to support multitasking, it needs to perform context switching periodically, which requires hardware resources such as timers.

The source interrupts program execution. When the timer interrupt is generated, the processor will perform OS task scheduling in exception handling,

At the same time, OS maintenance will be carried out. There is a simple timer called SysTick in the Cortex-M0 processor,

Used to generate periodic interrupt requests.

SysTick is a 24-bit timer and counts down. After the timer counts down to 0, it will reload a

A programmable value, and a SysTick exception (the exception number is 15) is generated at the same time, the exception event will cause

The execution of SysTick exception handling, this process is part of the OS.

For systems that do not require OS, the SysTick timer can also be used for other purposes, such as timing, timing, or

Tasks that need to be executed periodically provide interrupt sources. The generation of SysTick exception is controllable. If the exception is prohibited,

However, you can use the SysTick timer in a polling method, such as checking the current count value or polling the overflow flag.

27.2 Setting up SysTick

Since the reload value and current value of the SysTick timer are undefined at reset, in order to prevent abnormal results

As a result, the configuration of SysTick needs to follow a certain process:

Step1: Configure SysTick->CTRL. ENABLE to 0 to disable SysTick.

Step2: Configure SysTick->CTRL. CLKSOURCE and SYSTICK_CR[27:25], select SysTick

Clock source.

Step3: Configure SysTick->LOAD and select the overflow period of SysTick.

Step4: Write any value to SysTick->VAL, clear SysTick->VAL and SysTick->CTRL.

COUNTFLAG.

Step5: Configure SysTick->CTRL. TICKINT to 1, enable SysTick interrupt.

Step6: Configure SysTick->CTRL. ENABLE to 1, enable SysTick.

Step7: Read SysTick->CTRL in the interrupt service routine to clear the overflow flag.

Note: Systick overflow period is SysTick->LOAD+1, the configuration example is as follows:

Page 523

Clock source	SysTick->LOAD	Overflow period
RCH 4M	3999	1ms
XTL 32.768K	327	10.01ms

27.3 SysTick Register

address	name	CMSIS symbol	full name
0XE000E010	SYS_CSR	SysTick->CTRL	SysTick control and status register
0XE000E014	SYS_RVR	SysTick->LOAD	SysTick reload value register
0XE000E018	SYS_CVR	SysTick->VAL	SysTick current value register

27.3.1 SysTick control and status register (CTRL)

Bit	symbol	Function description	type	Reset value
31:17	Reserved	-	-	-
16	COUNTFLAG	Systick timer overflow flag 1: The Systick timer underflows. 0: The Systick timer does not overflow. Read this register to clear the COUNTFLAG flag	RO	0
15:3	Reserved	-	-	-
2	CLKSOURCE	SysTick clock source selection 1: Core clock (HCLK) 0: Reference clock, determined by SYSTICK_CR[27:25]	R/W	0
1	TICKINT	SysTick interrupt enable 1: Enable interrupt 0: Disable interrupt	R/W	0
0	ENABLE	SysTick timer enable 1: Enable SysTick 0: Disable SysTick	R/W	0

27.3.2 SysTick reload register (LOAD)

Bit	symbol	Function description	type	Reset value
31:24	Reserved	-	-	-
23:0	RELOAD	SysTick timer reload value	RW	-

Page 524**27.3.3 SysTick current value register (VAL)**

Bit	symbol	Function description	type	Reset value
31:24	Reserved	-	-	-
23:0	CURRENT	Read this register to get the current count value of the SysTick timer Write any value to this register, clear this register and COUNTFLAG	RW	-

28 Appendix B Document Conventions

28.1 List of Register-Related Abbreviations

The following abbreviations are used in the register description:

RW	Read and write, software can read and write these bits.
RO	Read only, software can only read these bits.
WO	Write only, software can only write to this bit. When this bit is read, invalid data will be returned.
W1	Write only 1, the hardware will automatically clear to 0, writing 0 is invalid
R0W1	The software reads this bit as 0, and writes 1 to clear the bit. Writing 0 has no effect on the value of this bit.
RW0	Software can read and write this bit, writing 1 is invalid, writing 0 clears
R1W0	The software reads this bit as 1, and writes 0 to clear the bit. Writing 1 has no effect on the value of this bit.
RC	Software can read this bit. When this bit is read, it will be cleared automatically. Writing "0" has no effect on the value of this bit.

Res, Reserverd are reserved bits and must be kept at reset values.

28.2 Glossary

This section briefly introduces the definitions of acronyms and abbreviations used in this document:

Word : 32-bit data.

Half Word : 16-bit data.

Byte : 8-bit data.

IAP (Programming in Application): IAP means that the Flash of the microcontroller can be reprogrammed during the running of the user program.

New programming.

ICP (In-Circuit Programming): ICP means that the JTAG protocol can be used when the device is installed on the user application circuit board.

SWD protocol or bootloader program the Flash of the microcontroller.

AHB : Advanced high-performance bus.

APB : Low-speed peripheral bus.

DMA : Direct memory access.

TIM : Timer

Version history & contact information

Version	Revision date	Summary of revisions
---------	---------------	----------------------

Rev1.1 2018/5/4	The version is updated, the Flash data is revised, and the description in Chapter 4 is revised.
Rev1.2 2018/5/23	Modified the clock switching process and added PCA comparison capture function module settings.
Rev1.3 2018/11/1	Supplement the description of function modules in Chapter 1, add the description of pin configuration and functions in Chapter 2, supplement the description of FLASH operation in Chapter 9, modify Chapter 28 Electrical Characteristic Parameters, add Chapter 29 and Chapter 31.
Rev1.4 2018/11/26	Modify the name: UART2→LPUART. Add "Notes" to Sections 2.1 and 2.2.
Rev1.5 2019/2/22	Revise the following data: ① ADC characteristics ② ESD characteristics ③ ECFLASH minimum in memory characteristics ④ Increase package size ⑤ AVCC/AVSS is added to the pin configuration diagram.
Rev1.6 2019/7/5	Correct the following data: ①correct UID address ②correct programming mode ③update ④
Rev1.7 2019/12/13	Correct the following data: ①Typical application circuit diagram ②LVD block diagram ③Bit register ④ I₂C , serial ⑤ LPUART ⑥
Rev1.8 2020/1/17	Amend the following data: ①Add note item ②Step8 of 17.5.1 and 17
Rev1.9 2020/3/6	Amend the following data: ①Add note item ②Step8 of 17.5.1 and 17
Rev2.0 2020/4/30	Correct the following data: ① ADC characteristics increase the accuracy of V _{IL} /V _{IH} ② Correction of clerical errors in the external clock source characteristics ③ Internal clock source characteristics Medium RCL oscillator accuracy.
Rev2.1 2020/5/29	Correct the following data: ① 17. Step2 and Step3 are added in 5 ② I ₂ Cx is changed to I ₂ C.
Rev2.2 2020/7/31	Amend the following data: ① Add TIM timer characteristics and communication interface section ② EFT level ③ Internal in general working conditions AHB/APB clock frequency ④ Correction of clerical errors ⑤ Input characteristics in port characteristics—ports P0, P1, P2, P3, V _{IL} and V _{IH} in RESET The value of V _{IL} .
Rev2.3 2020/9/30	Correct the following data: ① 7. 8 . 4 update clerical errors ② clock system description ③ RCH oscillator accuracy in the internal clock source characteristics ④ V _{IL} and V _{IH} of RESETB pin characteristics ⑤ Add SPI characteristics.
Rev2.31 2020/12/31	Delete product features, pin configuration, package information, etc. (for related information, please refer to the latest data sheet), and modify the statement.

If you have any comments or suggestions during the purchase and use process, please feel free to contact us .

Email: mcu@hdsc.com.cn

Website: <http://www.hdsc.com.cn/mcu.htm>

Mailing address: Floor 10, Building A, 1867 Zhongke Road, Pudong New Area, Shanghai

Post Code: 201203