Werner **B**als
HARDWARE & **S**OFTWARE
Wielinger Str. 20
D82340 Feldafing
Tel.:+49 8157 / 900491
Fax: +49 8157 / 900492
email: wernerb@cube.net

General description of OS9000 on MPC555 by BALS HARDWARE & Software

General

Table of Contents:

General 1
Table of Contents: 1
Contact information: 1
Basic features of OS9 on MPC555 2
Example include files 3
Example include file of internal mpc555 SIU 3
Example include file for serial interface QSM 5
Example init process for TOUCAN 6

Contact information:

Werner Bals
BALS HARDWARE & SOFTWAER
Wielinger Str. 20
82340 Feldafing
Germany
Tel.:+49 8157 900491 Fax:+49 8157 900492
mailto:wernerb@cube.net

Basic features of OS9 on MPC555

1. Full implemented .h-structure of processor internals
2. Full implemented .h structure for all internal peripherals like Serial interface, RAM, FLASH, ADC, CAN, TPU, MIOS, SCI, SPI, ...
3. full support of internal serial peripherals for standard os9 interface (term)
4. complete os9000 modules for irq, vector, timer
5. using internal decrementer for fast ticker
6. full support of mpc555 internal ram
7. init process initializing all internal features like chip select, bus interface
8. Coreboot-SW for booting without any additional software
9. Board support package fitting to standard PowerPC support package structure (Hawk 2.0 / PPC1.4)
10. HW recommendation list to fit to OS9000 requirements (vectors in RAM, external RAM and ROM, …)
11. Flash task to move OS9000-modules from module directory to internal flash of MPC555

The MPC555 is the first PowerPC – device with internal Flash. It also contains all automotive peripherals like Serial, CAN, Timer, …
The source files contains more than 75KB of include files !
Due to the fixed memory layout and the vector table requirements BALS HW & SW found a special solution for OS9000.
The coreboot created by BALS HW & SW can boot directly without additional Software.
The SIU IRQ module supports all internal peripherals.
The HW recommendation list contains a description to connect external peripherals (f.e. ethernet controller) to use correct bus access and irq model.

Example include files

Example include file of internal mpc555 SIU

```
#if !defined(_SIU555_H)
#define _SIU555_H

/*-------------------------------------------------------------------\
|  Definitions for the MPC555-SIU (System Interface Unit)             |
|  Edition History                                                    |
|                                                                     |
|  #    Date      Comments                                  By       |
|  --  --------  ----------------------------------------  ---  |
|  01  96/10/21  Created from siu505.h                       WB   |
\-------------------------------------------------------------------*/

#define SIUBASE       IMMRBASE+0x002fc000   /* Base address of SIU */

typedef struct siu555 {
       volatile unsigned longSIUMCR;       /* Configuration Register */
       volatile unsigned longSYPCR;        /* System Protection Control Register */
       long                                reserv0;
       volatile unsigned long       SWSR;         /* Software Service Register */
       volatile unsigned longSIPEND;       /* Interrupt Pending */
       volatile unsigned longSIMASK;       /* Interrupt Mask Register */
       volatile unsigned longSIEL;         /* Interrupt edge level Register */
       volatile unsigned longSIVEC;        /* Interrupt vector */
       volatile unsigned longTESR;         /* Transfer error status register */
       volatile unsigned longSGPIODT;      /* USIU general purpus I/O Data register*/
.............
       long                                reservB[4];
       volatile unsigned longPISCRIK;      /* pit status and control key */
       volatile unsigned longPITCK;        /* pit count key */
       long                                reservC[14];
       volatile unsigned longSCCRK;        /* system clock control key */
       volatile unsigned longPLPRCRK;      /* pll low power and reset control reg. key */
       volatile unsigned longRSRK;         /* reset status register key */

} SIU555; /* end struct siu505 */

/*-----------------------------------------------------\
! Bit definitions for the different registers of SIU555 !
\-----------------------------------------------------*/

/* SIUMCR - SIU Module Configuration Register (see MPC555 Manual, Page 6-19) */
#define EARB          0x80000000            /* Ext. Arbitration */
#define EARP          0x70000000            /* ext. arb. priority */
#define DSHW          0x00800000            /* show data cycles */
#define DBGC          0x00600000            /* debug pin conf. */
#define DBPC          0x00100000            /* debug port pin conf. */
#define ATWC          0x00080000            /* add. write type anebla conf. */
#define GPC                  0x00060000           /* pin conf. */
#define DLK                  0x00010000           /* debug register clock */
#define SC            0x00006000            /* single chip select */
#define RTCX          0x00001000            /* reset conf. timer expired */
#define MLRC          0x00000C00            /* multi level res. contr. */


/* SYPCR - system protection control register */
#define SWP                  0x00000001           /* sw watchdog timer prescaled */
#define SWRI          0x00000002            /* sw watchdog causes reset */
#define SWE                  0x00000004           /* sw watchdog enable */
#define BME                  0x00000008           /* bus monitor enable */
#define BMT                  0x0000ff00           /* max. bus monitor time */

/* SWSR - Software Watchdog Service Register */
#define SWSR_TRIG1    0x556C                /* first value to write to SWSR for triggering */
#define SWSR_TRIG2    0xAA39                /* second value to write to SWSR for triggering */

/* SIPEND - SIU interrupt pendign register */
#define IRQ0          0x80000000
#define LVL0          0x40000000
#define IRQ1          0x20000000
#define LVL1          0x10000000
```

```
#define IRQ2            0x08000000
#define LVL2            0x04000000
#define IRQ3            0x02000000
#define LVL3            0x01000000
#define IRQ4            0x00800000
#define LVL4            0x00400000
#define IRQ5            0x00200000
#define LVL5            0x00100000
#define IRQ6            0x00080000
#define LVL6            0x00040000
#define IRQ7            0x00020000
#define LVL7            0x00010000

/* SIMASK - SIU interrupt mask register */
#define IRM0            0x80000000
#define LVM0            0x40000000
....
.....
....
```

Example include file for serial interface QSM

```
#if !defined(_QSMCM_H)
#define _QSMCM_H
/*-------------------------------------------------------------------\
|                                                                    |
|  Definitions for the MPC555 QSM (Queued Serial Module)             |
|                                                                    |
|                                                                    |
|  Edition History                                                   |
|                                                                    |
|  #    Date      Comments                                  By       |
|  --  --------  ----------------------------------------------------|
|  01  98/02/25  Created from "qsm16.h"                      WB    |
|                                                                    |
\-------------------------------------------------------------------*/
#define QSMCM_BASE IMMRBASE+0x00305000
/* This is the structure of the registers that are associated with QSM.
   It is defined as offsets from QSMCM_BASE. */

typedef struct qsmcm {
       volatile unsigned short      QSMCR;  /* Configuration register */
       volatile unsigned short      QTEST;  /* QSM Test register, for factory test only */
       volatile unsigned short      QDSCI_IL;      /* dual QSM IRQ level register (MSB only)
*/
       volatile unsigned short QSPI_IL;     /* QSPI interupt level register (LSB only) */
       /********* Asynchronous Serial Port 1 **********/
       volatile unsigned short      SCC1R0;/* Baud Rate              ---\              */
       volatile unsigned short      SCC1R1;/* Baud Rate              ---\              */
       volatile unsigned short      SCS1SR;/* Status Register      |--  UART  ( /TERM )*/
       volatile unsigned short      SC1DR;  /* Data Register          ---/              */
       short                                notused1[2];
       volatile unsigned short      PORTQS;/* Port data register (LSB only) */
       volatile unsigned short DDRQS;       /* Pin assignment (MSB) and Data direction (LSB)
register */
       /******** High Speed Synchronus Port ********/
       volatile unsigned short      SPCR0;  /* Control Register 0 */
       volatile unsigned short SPCR1;       /* Control Register 1 */
       volatile unsigned short SPCR2;       /* Control Register 2 */
       volatile unsigned short      SPSR;   /* Control Register 3 (MSB) and Status Register
(LSB) */
       /********* Asynchronous Serial Port 2 **********/
       volatile unsigned short      SCC2R0;/* Baud Rate              ---\              */
       volatile unsigned short      SCC2R1;/* Baud Rate              ---\              */
       volatile unsigned short      SCS2SR;/* Status Register      |--  UART  ( /TERM )*/
       volatile unsigned short      SC2DR;  /* Data Register          ---/       */
       /* qsci controls */
       volatile unsigned short      QSCI1CR;
       volatile unsigned short      QSCI1SR;
       volatile unsigned short      SCTQ[ 16];     /* transmit queue locations */
       volatile unsigned short      SCRQ[ 16];     /* transmit queue locations */
       short                               notused[ 106];
       volatile unsigned short      RR[32];/* 32 half word Receive RAM */
       volatile unsigned short      TR[32];/* 32 half word Transmit RAM */
       volatile unsigned charCR[32];/* 32 Byte Command RAM */
} QSMCM;   /* end struct qsmcm */

/*##########################################################################*
 ************  These are individual field masks and bits for the  *******
 ************    registers that are defined above.                *******
 *##########################################################################*/

/********** SCI - Serial Communications Interface -- UART ************
 **  These registers make up the serial interface for the TERM device. */

/*********** QMCR ********************/
#define QSMSTOP 0x8000       /* QSM Stop Enable */
#define QSMFRZ1 0x4000       /* Freeze1 - determines action when FREEZE is asserted. */
#define QSMSUPV 0x0080       /* Supervisor/Unrestricted (1 = supervisor state only access) */
#define QSMIARB 0x0006       /* Interrupt Arbitration Identification Number */

/*********** QDSCI_IL **********/
#define ILQSPI 0x001f        /* mask for the QSPI level bits */
……
```

Example init process for TOUCAN

```
/* ------------------------------------------------------------------- */
void _init_can(TOUCAN *psTou) /* init TOUCAN controller in  MPC555*/
{
unsigned int nCount;


/* init operation modes ------------------------------ */
/* 250kBaud */
psTou->CANCTRL0 = (0<<RXMODE) | (0<<TXMODE) | (6<<PROPSEG);

/* bit clock is 5 MHz == 25MHz / 5; seg1 = seg2 = 1,2 us */
psTou->CANCTRL2 = (4<<PRESC) |(3<<RJW) | (5<<PSEG2) | (5<<PSEG1);
psTou->CANICR = 0; /* no irqs */


/* init message buffers for normal operation
 * */

/* first init receive buffers 0..11 */
for (nCount = 0; nCount < 11; nCount++)
       {
       psTou->MSB[ nCount].CONTR_STAT = (R_NOTACTIV<<TOUCODE);  /* receive 0 */
       }

/* then init receive ids */
psTou->MSB[ 0].ID_HIGH = ((0x200 | (WR_ID<<4))<<ID18);     /*  */

psTou->MSB[14].ID_HIGH = ((0x500 | (WR_ID<<4))<<ID18);     /*  */

/* ----------- transmit buffers ------------------------------------- */
/* then init transmit buffer 1 */
psTou->MSB[ 1].ID_HIGH = ((0x600 | WR_ID)<<ID18);   /* upload */
psTou->MSB[ 1].CONTR_STAT = (T_NOTREADY<<TOUCODE);  /* receive 11 */


/* enable msb 0 to receive commands */
psTou->MSB[ 0].CONTR_STAT = (R_EMPTY<<TOUCODE); /* receive commands */

/* ------------------------------------------------------------------- */
psTou->IFLAG = 0;            /* clear rec. and transmit flags */
psTou->ECTR = 0;             /* clear error counters */

/* clear all existing error */
nCount = psTou->ESTAT;/* firs read error status */
psTou->ESTAT = 0;

psTou->CANMCR  = TOUFS_enable & ~TOU_halt & ~TOUSV_only; /* start */

}    /* end of _init_can() */
```