TAMPEREEN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma
Tietokonetekniikan suuntautumisvaihtoehto

Sakari Mikkola

P89LPC935-MIKROKONTROLLERIN OMINAISUUDET JA KÄYTTÖ

Tutkintotyö, joka on jätetty opinnäytteenä tarkastettavaksi insinöörin tutkintoa varten Tampereella 12.5.2006

Työn valvoja: Yliopettaja Kai Poutanen

Tekijä: Sakari Mikkola
Työn nimi: P89LPC932-mikrokontrollerin ominaisuudet ja käyttö

Päivämäärä: 2.6.2006
Sivumäärä: 31
Hakusanat: P89LPC935, mikrokontrolleri
Koulutusohjelma: Tietotekniikka
Suuntautumisvaihtoehto: Tietokonetekniikka

Työn valvoja: Yliopettaja Kai Poutanen

Nykyteknologia sisältää yhä enemmän peruskomponenttien lisäksi kehittyneempiä komponentteja kuten erilaisia mikropiirejä ja mikrokontrollereita. Erityisesti mikrokontrollerit ovat tärkeitä ja niitä onkin lähes kaikissa elektronisissa laitteissa. Esimerkkeinä voidaan mainita autot, tietokoneet, kaikki kodinelektroniikka ja niin edelleen.

Mikrokontrollerien käyttö on yleistynyt, koska ne ovat ominaisuuksiltaan monipuolisia ja niiden ohjelmointi ja käyttö on melko yksinkertaista. Mikrokontrollereita käytetään yleensä pienien ja keskisuurten kokonaisuuksien ohjaamiseen. Niillä voidaan kuitenkin toteuttaa myös laajempia kokonaisuuksia, jotka koostuvat sadoista mikrokontrollereista.

Mikrokontrollerien perusrakenne on samankaltainen. Se koostuu muutamasta peruslohkosta, joita ovat keskusyksikkö, muisti, I/O-piiri, sarjaliikennepiiri, väylä, kello, A/D-muunnin ja watchdog.

Peruslohkojen lisäksi Philipsin P89LPC935-mikrokontrolleri sisältää useita muita lohkoja, jotka lisäävät mikrokontrollerin ominaisuuksia ja käyttökohteita. Näistä voidaan mainita esimerkiksi analoginen komparaattori, jolla voidaan vertailla analogisia signaaleja ja ohjailla vertailutulosten perusteella mikrokontrollerin toimintoja.

P89LPC935-mikrokontrolleri on kiinnitetty Keilin MCB900-testialustaan, jolla voidaan testata mikrokontrollerin toimintaa. Testialustassa on sarjaliikenneportti, jonka avulla se voidaan liittää esimerkiksi tietokoneeseen. Tietokoneelle asennettavan Keilin µVision-ohjelmiston avulla voidaan kirjoittaa mikrokontrollerille ohjelmia, kääntää ja linkata ne, ladata ohjelmat mikrokontrollerin muistiin ja ajaa ne. Lisäksi ohjelmistolla voidaan simuloida kirjoitetut ohjelmat ilman, että niitä tarvitsee ladata mikrokontrollerin muistiin.

Author:	Sakari Mikkola
Name of the thesis:	The attributes of P89LPC932-microcontroller
Date:	2.6.2006
Pages:	31
Keywords:	89LPC935, microcontroller
Degree programme:	Computer systems engineering
Specialization:	Computer engineering
Supervisor:	Principal lecturer Kai Poutanen

Modern technology includes except ordinary electronic components as well as many kind of integrated circuits and microcontrollers. Especially the microcontrollers are very common components in all kind of electronic devices. Microcontrollers are used for example in cars, computers, calculators, microwave ovens etcetera.

Microcontrollers are very common components because they have many attributes and they are user friendly. Microcontrollers are mostly used in quite small and simple applications but they can be also used in complex systems. Then the system can include hundreds of microcontrollers.

The basic structure of all microcontrollers is similar. The basic blocks are CPU, memory, I/O, A/D-converter, bus, serial interface and watchdog.

In addition of the basic structure Philips P89LPC935-microcontroller includes many additional blocks that make the microcontroller more versatile. For example an analog comparator is one of the additional blocks. The analog comparator can be used to compare incoming analog signals and control the microcontroller according the comparison results.

P89LPC935-microcontroller is connected to Keil MCB900-testboard. The MCB900-testboard is used to test and to monitor the microcontroller. The testboard includes a serial communication port what is used to connect the testboard to a computer. In addition of the testboard Keil µVision-software is needed to use the testboard. With the µVision-software it is possible to write programs, compile and link them, load the programs to the microcontroller or simulate the programs without loading them to the microcontroller.

ALKUSANAT

Tämä tutkintotyö on tehty kevään 2006 aikana itsenäisesti ilman erillistä ohjaajaa. Aiheen tähän työhön antoi Tampereen ammattikorkeakoulun yliopettaja Kai Poutanen, joka toimi myös työn valvojana.

Haluan kiittää yliopettaja Kai Poutasta, joka antoi aiheen työhön. Erityisesti haluan kiittää perhettäni ja ystäviäni, jotka ovat auttaneet ja kannustaneet minua.

Tampereella 2. kesäkuuta 2006

_____
Sakari Mikkola

SISÄLLYSLUETTELO

KÄYTETYT LYHENTEET JA TERMIT

ADC, A/D	Analog-to-Digital Converter, Analog/Digital. Piiri, jolla muutetaan analoginen signaali digitaaliseen muotoon.
BOOTSTAT.0	Bitti, joka ilmoittaa uudelleenkäynnistyksen tilan.
BRGR	Baud Rate Generator Register. Erikoisrekisteri, joka sisältää tiedonsiirtonopeuteen liittyviä esiohjelmoituja arvoja.
CCLK	Computer Clock. Keskusyksikön kellojakso.
CCU	Capture/Compare Unit. Yksikkö, joka tallettaa laskurin arvoja sisääntulevan signaalin muutoskohdissa ja vertaa niitä referenssiarvoihin.
CCUCLK	Capture/Compare Unit Clock. CCU-yksikön kellosignaali.
CMF1, CMF2	Komparaattorien 1 ja 2 ohjausrekisterit, joihin talletetaan esimerkiksi ulostuloissa tapahtuvat muutokset.
CNF	Capture Noise Filter. Signaalinkaappauksessa käytettävä yksinkertainen suodatin.
CPU	Central Processing Unit. Esimerkiksi mikrokontrollerin keskusyksikkö.
DAC, D/A	Digital-to-Analog Converter, Digital/Analog. Piiri, jolla muutetaan digitaalinen informaatio analogiseksi.
DATA	Datamuisti. Käyttömuisti, jota voidaan osoittaa sekä suoraan että epäsuorasti.
EEPROM	electrically erasable programmable read only memory. Sähköisesti pyyhittävä muisti, jota voidaan vain lukea.
I^2C	Elektroniikassa käytettävien piirien kaksisuuntainen ohjaus- ja tiedonsiirtoväylä.
IC	Integrated Circuit. Mikropiiri.
IDATA	Indirect Data. Epäsuora datamuisti, jota voidaan osoittaa vain epäsuorasti.
I/O	Input/Output. Esimerkiksi komponentti tai piiri, joka huolehtii kaksisuuntaisesta dataliikenteestä.

ISP	In-System Programming. Mahdollistaa laitteen ohjelmoinnin samanaikaisesti sen ollessa yhteydessä sovelluksen kanssa.
KBCON	Keypad Interrupt Control Register. Näppäimistökeskeytyksen ohjausrekisteri.
KBI	Keypad Interrupt. Näppäimistön tai väyläosoitteen tunnistamisen aiheuttama keskeytys.
KBIF	Keypad Interrupt Flag. Näppäimistökeskeytysrekisterin lippubitti.
KBMASK	Keypad Interrupt Mask register. Näppäimistökeskeytyksen maskirekisteri.
KBPATN	Keypad Pattern Register. Näppäimistökeskeytyksen rakennerekisteri, mihin portti 0:ssa tapahtuvia muutoksia verrataan.
LCD	Liquid Crystal Display. Nestekidenäyttö.
LSB	Least Significant Bit. Bittijonon vähiten merkitsevä bitti.
MISO	Master In Slave Out. SPI-lohkon tiedonsiirtosuunta, jossa data liikkuu orjalaitteesta isäntälaitteeseen.
MOSI	Master Out Slave In. SPI-lohkon tiedonsiirtosuunta, jossa data liikkuu isäntälaitteesta orjalaitteeseen.
OSCCLK	Oscillator Clock. Oskillaattorin kellojakso.
PCLK	Peripheral Clock. Kellosignaali, jota käytetään useimmissa mikrokontrollerin oheislaitteissa.
PCON	Power Control Register. Tehonvalvontarekisteri, jolla ohjataan esimerkiksi mikrokontrollerin tiedonsiirtoa.
PLL	Phase-Locked Loop. Vaihelukittu silmukka.
POF	Power-off-reset flag. Lippu-bitti, joka osoittaa mikrokontrollerin käyttöjännitteen kytkennän eri vaiheet.
PSW	Program Status Word. Prosessorin tilarekisteri.
PWM	Pulse Width Modulation. Pulssinleveysmodulaattori, jolla voidaan muokata erilaisia pulssisuhteita.

RAM	Random Access Memory. Muisti on käyttömuistia, eli sitä voidaan kirjoittaa ja lukea.
RB8	Sarjaliikennerekisterin bitti, johon talletetaan datansiirtoon liittyviä arvoja.
RPE	Reset Pin Enable. Mikrokontrollerin resetointinastan salliminen.
RS232	Pisimpään käytössä ollut tietoliikenneportti, joka on tarkoitettu lähinnä kahden laitteen liittämiseksi toisiinsa.
RSTSRC	Reset Source Register. Järjestelmän palautusrekisteri, joka sisältää tietoa esimerkiksi mikrokontrollerin palautusparametreista.
RTC	Real Time Clock. Reaaliaikakello.
RTCF	Real Time Clock Flag. Reaaliaikakellon lippubitti, joka asetetaan laskurin saavuttaessa arvon 0.
RXD	Received Exchange Data. Nasta tai linja vastaanotettavalle datalle.
SBUF	Serial Port Data Buffer register. Tiedonsiirrossa käytettävä puskurointirekisteri, johon voidaan tallettaa lähetystä odottavaa tietoa.
SCL	Serial Clock Input/Output. I^2C-väylän johto, jota pitkin kellosignaalit kulkevat.
SCON	Serial Port Control. Osa mikrokontrollerin erikoisrekisteriä, johon talletetaan datansiirtoon liittyviä arvoja
SDA	Serial Data. I^2C-väylän johto, jota käytetään laitteidenvälisessä tiedonsiirrossa.
SFR	Special Function Registers. Mikrokontrollerin rekisteri, joka sisältää tiettyjä erikoistoimintoja.
SM0	SCON-rekisterin eniten merkitsevä bitti.
SPI	Serial Peripheral Interface. Liityntäpinta, jolla erilaisia lisälaitteita voidaan liittää mikrokontrolleriin.
SPICLK	SPI Clock. SPI-sarjaliikennelohkon kellosignaali.
SS	SPI Slave Select. SPI-lohkon orjavalintasignaali.

SSTAT	Tilarekisteri, johon talletetaan tietoa esimerkiksi mikrokontrollerin tiedonsiirrosta.
TB8	Sarjaliikennerekisterin bitti, johon talletetaan datansiirtoon liittyviä arvoja.
TOR	Time-Out Register. Pulssinleveysmoduloinnissa käytettävä ohjausrekisteri.
TRIM	Trimming. Rekisteri, jolla säädetään RC-oskillaattorin parametrejä P89LPC935-mikrokontrollerissa.
TXD	Transmit Exchange Data. Nasta tai linja lähetettävälle datalle.
UART	Universal Asynchronous Receiver/Transmitter. Sarjaliikennepiiri, joka muuttaa rinnakkaismuotoisen datan sarjamuotoiseksi ja päinvastoin.
UCFG1	User Configuration Register 1. Mikrokontrollerin rekisteri, jonka sisältöä muuttamalla käyttäjä voi vaikuttaa mikrokontrollerin toimintaan.
USB	Universal Serial Bus. Sarjaväyläarkkitehtuuri, jolla voidaan liittää erilaisia laitteita esimerkiksi tietokoneeseen.
V_{dd}	Elektroniikkapiirien, esim. mikrokontrollerien ja vahvistimien käyttöjännitenasta.
V_{RAM}	Minimijännite, jolla RAM-muistin sisältö voidaan taata.
Watchdog	Vahtikoira. Käytetään mikrokontrollereissa mahdollisissa vikatilanteissa.
XDATA	Auxiliary Data Memory. Lisämuisti.
XTAL1/XTAL2	Mikrokontrollerin liityntänastat, joihin liitetään ulkoinen kide.

1 JOHDANTO

Mikrokontrollerien käyttö on kasvanut räjähdysmäisesti vuodesta 1971, jolloin markkinoille tuli ensimmäinen kaupalliseen tarkoitukseen kehitetty mikroprosessori. Tuolloin ensisijainen käyttötarkoitus oli laskukone. Nykyään mikrokontrollereita on lähes kaikissa teollisuudessa käytettävissä elektronisissa laitteissa. Kotitalouksissakaan ei enää ole montaa sellaista laitetta, joka ei sisältäisi vähintään yhtä mikrokontrolleria. Hyvänä esimerkkinä ovat muun muassa mikroaaltouunit, kahvinkeittimet, kelloradiot ja kaikki muut tutut kotitalouslaitteet.

Tämän tutkintotyön tavoitteena on tutustua Philipsin P89LPC935-mikrokontrollerin toimintaan, Keilin MCB900-testialustaan sekä µVision-ohjelmistoon.

MCB900-testialusta on suunniteltu Philipsin P89LPC93X-tuoteperheelle ja se toimii adapterina mikrokontrollerin ja tietokoneen välillä. µVision-ohjelmistolla testialusta voidaan yhdistää suoraan tietokoneeseen joko RS232- tai USB-liitännällä. Ohjelmiston avulla voidaan esimerkiksi ladata suoritettavat ohjelmat mikrokontrolleriin, käydä ajettavaa ohjelmaa lävitse käsky käskyltä ja seurata mikrokontrollerin rekisterisisältöjä monitorointiohjelman avulla. Lisäksi ohjelmistolla voidaan simuloida erilaisia ohjelmia ilman, että ohjelmia tarvitsee ladata mikrokontrollerin muistiin.

Luvussa 2 selvitetään käsitteet mikrokontrolleri ja mikroprosessori sekä tarkastellaan lähemmin mikrokontrollerien yleistä rakennetta. Luvussa 3 luodaan kevyt katsaus Philipsin P89LPC935-mikrokontrollerin yleisiin ominaisuuksiin ja lohkokaavioon. Luvussa 4 käydään spesifioidusti läpi P89LPC935-mikrokontrollerin rakenne ja selvitetään eri lohkojen toiminta osana mikrokontrolleria. Luvussa 5 puolestaan tutustutaan A/D-muuntimeen ja sen toimintaa. Lopuksi tutustutaan MCB900-testialustaan, µVision-ohjelmistoon sekä mikrokontrollerin testauksessa käytettyihin testiohjelmiin luvussa 6.

2 MIKROKONTROLLERIT JA MIKROPROSESSORIT

Mikrokontrollereilla on keskeinen osa nykypäivän tekniikassa. Niitä löytyy lähes kaikista teknisistä laitteista, kuten esimerkiksi LCD-näytöistä, mikroaaltouuneista, näppäimistöistä ja autoista. Mikrokontrollereilla voidaan toteuttaa erilaisia ajastinpiirejä, A/D-muunnoksia, signaalivertailuja ja moottorinohjauksia. Käyttökohteita on lähes rajattomasti.

Mikrokontrolleria ei pidä sekoittaa mikroprosessoriin. Yhteistä näille komponenteille on se, että kaikki mikrokontrollerit sisältävät mikroprosessorin. Mikroprosessori on kuitenkin vain yksi osa kokonaisuutta, johon pitää liittää muita ulkoisia komponentteja toiminnan takaamiseksi. Tällaisia komponentteja ovat esimerkiksi muistit ja datansiirrosta huolehtivat komponentit. Mikrokontrolleri on puolestaan toimiva kokonaisuus, mikä ei tarvitse välttämättä kuin ulkoisen jännitelähteen toimiakseen. Mikrokontrolleriin on integroitu kaikki tarvittavat komponentit, kuten juuri muistit ja I/O-nastat mitkä huolehtivat datansiirrosta. [2]

Kyseisten komponenttien käyttökohteet eroavat toisistaan. Mikroprosessoreita käytetään yleensä suurta laskentatehoa vaativissa sovelluksissa, kuten esimerkiksi tietokoneissa. Mikrokontrollereita puolestaan käytetään pienemmissä kokonaisuuksissa, kuten esimerkiksi roboteissa ja tiedonsiirtolaitteissa.

Mikrokontrollerien historia ulottuu vuoteen 1971, jolloin Intel julkaisi maailman ensimmäisen mikroprosessorin. Intel 4004 -mikroprosessori oli 4-bittinen. Siinä oli noin 2300 transistoria ja sen kellotaajuus oli maksimissaan 760 kHz. Se oli ensimmäinen IC-piiri, johon oli integroitu sekä ohjelma- että datamuisti. Lisäksi piiri sisälsi kuusitoista 4-bittistä rekisteriä ja 46 käskyn käskykannan. Mikroprosessori suunniteltiin alun perin käytettäväksi laskukoneissa. [3]

2.1 Yleinen rakenne [2]

Mikrokontrolleri koostuu useasta erillisestä lohkosta, jotka yhdessä muodostavat toimivan kokonaisuuden. Tärkeimpiä osia ovat CPU eli keskusyksikkö, muistilohko, I/O-lohko, väylä, sarjaliikennelohko, A/D-muunnin, kello ja watchdog-lohko. Jokaisella lohkolla on oma tehtävänsä, joita CPU ohjaa ja valvoo.

2.1.1 Keskusyksikkö

Keskusyksikkö eli CPU on yksi mikrokontrollerin keskeisimmistä lohkoista, mikä ohjaa koko mikrokontrollerin toimintaa. Vaikka mikrokontrollerit sisältävät aina erillisiä muistipiirejä, on keskusyksikössä myös sisäänrakennettua muistia. Tätä muistia kutsutaan rekistereiksi. Rekistereitä käytetään yleensä väliaikaisina muisteina

esimerkiksi erilaisia laskutoimituksia tehdessä. Keskusyksikkö kiinnittyy mikrokontrollerin data- ja osoiteväyliin sekä ohjainyksiköihin.

2.1.2 Muistilohko

Muistilohko on yksi mikrokontrollerin osa, minkä tärkein tehtävä on tallentaa dataa myöhempää käyttötarkoitusta varten. Yleisesti datantallennuksessa käytetään RAM-muistia, jota voidaan sekä lukea että kirjoittaa. Muistipiiri jakautuu eri muistiosoitteisiin, minkä perusteella tallennettu data pystytään tarvittaessa palauttamaan oikeasta paikasta.

Nykyisissä mikrokontrollereissa on yleensä myös EEPROM-muistia. Kyseinen muisti on sähköisesti ohjelmoitavaa lukumuistia. EEPROM-muisti sisältää yleensä kyseisen mikrokontrollerin tunnisteet ja mahdolliset ohjelmien versionumerot. Lisäksi muisti voi sisältää niin sanotut tehdasasetukset, joilla mikrokontrolleri voidaan palauttaa perustilaan.

2.1.3 Väylä

Väylä yhdistää mikrokontrollerin eri osat toisiinsa ja siirtää niiden välillä dataa. Väylä jaetaan kahteen eri osaan, osoite- ja dataväylään. Osoiteväylä yhdistää CPU:n ja muistipiirin keskenään. Osoiteväylän tehtävänä on välittää CPU:n antamat muistipaikkojen osoitteet muistipiirille. Dataväylä puolestaan yhdistää kaikki mikrokontrollerin yksittäiset lohkot toisiinsa. Väylä huolehtii mikrokontrollerin kaikesta sisäisestä tiedonsiirrosta.

Mikrokontrollerin sisäinen tiedonsiirto tapahtuu rinnakkaismuotoisena, joten silloin tarvitaan jokaista siirrettävää bittiä kohti oma fyysinen johto. Jos esimerkiksi dataväylä on 16-bittinen, tarvitaan 16 erillistä johtoa siirtämään tarvittava data.

2.1.4 I/O-lohko

I/O-lohkon tehtävänä on hoitaa mahdollinen tiedonsiirto mikrokontrollerin ulkopuolelle. Usein tämän ulkopuolisen kommunikoinnin hoitaa RS232-liityntä, mutta uusimmissa mikrokontrollereissa on jo yleisesti käytössä USB-liityntä mikrokontrollerin ja esimerkiksi tietokoneen välillä.

2.1.5 Sarjaliikennelohko

Sarjaliikennelohkon tehtävänä on muuttaa mikrokontrollerin sisäinen, rinnakkaismuotoinen data sarjamuotoiseksi. Kyseinen muunnos on varsin hyödyllinen silloin, kun mikrokontrolleri kommunikoi ulkopuolisen laitteen kanssa. Sarjamuotoisen datan siirrossa tarvitaan ainoastaan yksi johto lähtevälle ja yksi johto saapuvalle datalle.

Tällöin säästetään huomattavasti materiaalikustannuksissa ja mikrokontrollereista voidaan tehdä kompaktimman kokoisia.

2.1.6 A/D-muunnin

Mikrokontrollerien toimintaan liittyy usein ulkoisia antureja ja mittalaitteita. Koska useimmat anturit ovat analogisia, eivät ne voi suoraan kommunikoida mikrokontrollerin binäärisen käskykannan kanssa. Tällöin tarvitaan A/D-muunnin eli ADC-yksikkö, joka muuttaa mitta-anturin analogisen tiedon digitaaliseen muotoon. Muunnos voidaan tehdä myös toiseen suuntaan, tällöin kyseessä on D/A-muunnos.

2.1.7 Kello

Kello on yksi mikrokontrollerin tärkeimmistä osista. Se tahdistaa koko mikrokontrollerin toiminnan ja sen avulla saadaan tietoa esimerkiksi käskyjen suoritusnopeudesta ja eri toimintojen aiheuttamasta viiveestä. Kellotaajuus ilmaisee sen, montako käskyä CPU voi suorittaa sekunnissa. Yksinkertaisimmatkin laskutoimitukset vaativat useita käskyjä, joten CPU:n laskentanopeus on aina kellotaajuutta pienempi. Suurempi kellotaajuus merkitsee nopeampaa tietojen prosessointikykyä. Nykyään perusmikrokontrollereiden kellotaajuus on noin 5 – 50 MHz, kun ensimmäisen mikrokontrollerin kellotaajuus oli 760 kHz.

2.1.8 Watchdog

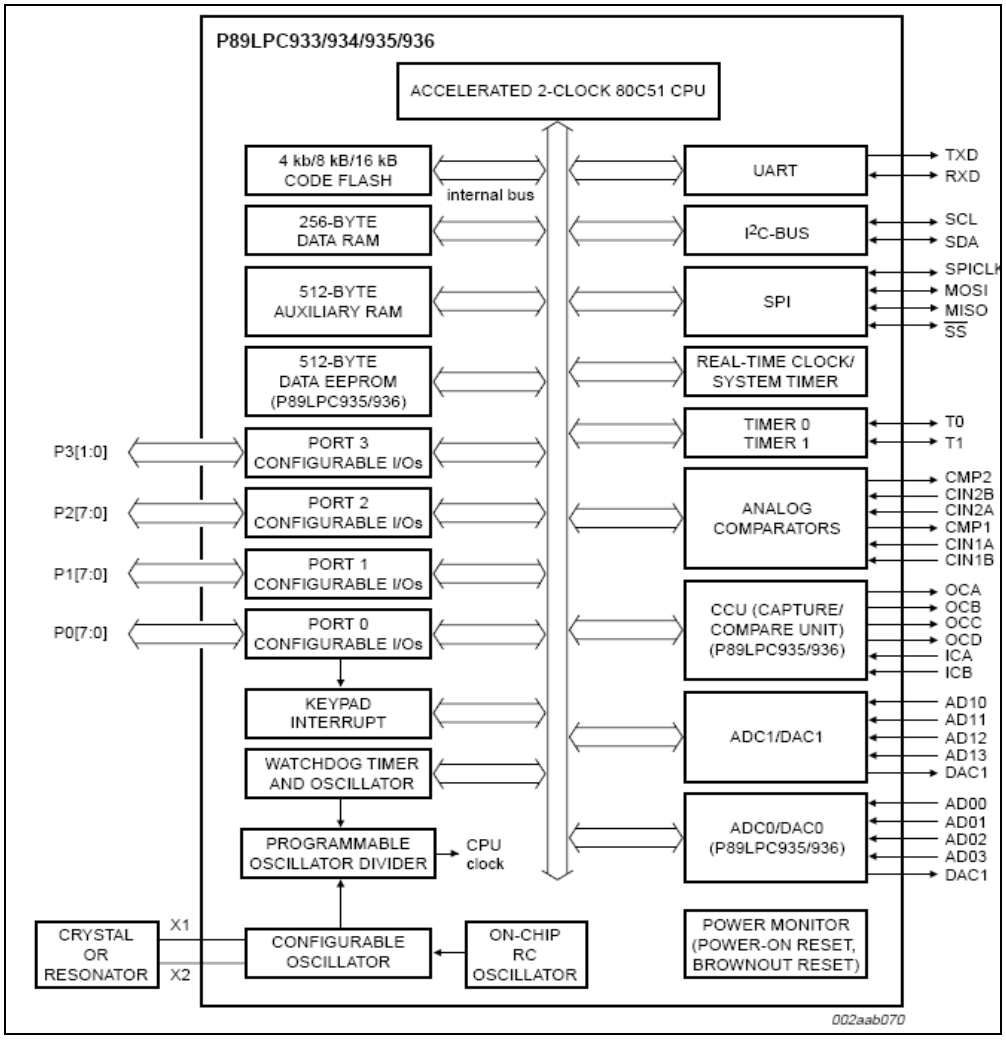
Watchdogin eli vahtikoirayksikön tarkoituksena on taata mikrokontrollerin virheetön toiminta, ja mahdollisten virheiden sattuessa suorittaa virheenkorjaus itsenäisesti. Watchdog-yksikkö sisältää laskurin, johon ajettava ohjelma kirjoittaa nollan aina kun ohjelma suoritetaan onnistuneesti. Mikäli ohjelma jostain syystä jumiutuu, ei nollan kirjautumista tapahdu. Tällöin watchdog-yksikkö resetoi mikrokontrollerin, ja ohjelman suoritus alkaa alusta.

3 P89LPC935-MIKROKONTROLLERI [1]

Philipsin P89LPC935-mikrokontrolleri on toteutettu yhdellä mikrosirulla, mikä perustuu korkean suorituskyvyn prosessoriarkkitehtuuriin. P89LPC935-mikrokontrollerin rakenne pohjautuu 80C51-piirisarjaan, mutta siihen on tehty monia muutoksia esimerkiksi piirin nopeuden parantamiseksi. Mikrokontrolleri käyttää 80C51-ydintä, missä on kaksi 16-bittistä ajastinta.

3.1 Mikrokontrollerin rakenne ja lohkokaavio

Kuvassa 1 on esitettynä P89LPC935-mikrokontrollerin lohkokaavio ja liityntänastat. Myöhemmissä luvuissa käsitellään kaikki mikrokontrollerin eri lohkot ja selvittämään niiden toimintaperiaate ja tehtävä mikrokontrollerissa.



Kuva 1. P89LPC935-mikrokontrollerin lohkokaavio [1]

4 P89LPC935-MIKROKONTROLLERIN TOIMINNALLINEN KUVAUS [1]

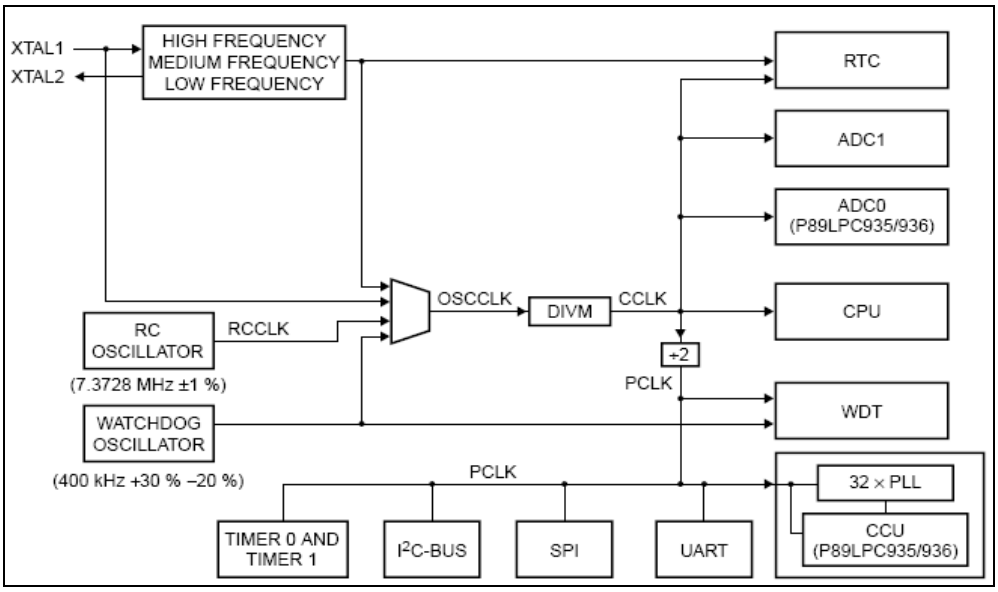
Mikrokontrollerien kehitys on ollut nopeaa aina 1970-luvulta lähtien, jolloin ensimmäinen mikrokontrolleri tuli markkinoille. Perusominaisuudet ovat lisääntyneet huomattavasti ja samalla mikrokontrollerien toiminnallinen nopeus on moninkertaistunut.

Tässä luvussa tarkastellaan Philipsin P89LPC935-mikrokontrollerin sisältämiä toiminnallisia ominaisuuksia.

4.1 Kellosignaalit

P89LPC935-mikrokontrollerissa on 23-bittinen järjestelmäkello (OSCCLK), johon sisältyy 7-bittinen esijakaja. Järjestelmäkelloa voidaan käyttää myös RTC-kellona. Lisäksi mikrokontrollerissa on kaksi 16-bittistä laskuria, joita voidaan käyttää kellosignaaleina.
Järjestelmäkellon asetuksia voidaan muuttaa ohjelmallisesti. Näin pystytään optimoimaan käytettävä kellosignaali kunkin käyttötarkoituksen mukaan sopivaksi. Myös CCLK-signaalista käytetään nimitystä järjestelmäkello.

Kuvassa 2 on kellosignaaliohjauksen lohkokaavio, mistä käy ilmi käytettävän järjestelmäkellon ohjaussignaalit. OSCCLK-signaali voidaan valita joko watchdog-oskillaattorilta, RC-oskillaattorilta, ulkoista kidettä käytävältä oskillaattorilta tai käyttää kokonaan ulkoista kellosignaalia. Watchdog- ja RC-oskillaattorit ovat mikrokontrolleriin kuuluvia sisäisiä osia.



Kuva 2. Kellosignaaliohjauksen lohkokaavio [1]

Ulkoista kidettä käytettäessä voidaan valita kide, jonka värähtelytaajuus asettuu 20 kHz:n ja 12 MHz:n väliin. Ulkoinen kide liitetään kuvassa 2 näkyvien nastojen XTAL1 ja XTAL2 väliin. Mikäli P89LPC935-mikrokontrollerin kellosignaalia halutaan käyttää ulkoisen laitteen kellosignaalina, se voidaan valita ohjelmallisesti. Tällöin lähtevä kellosignaali syötetään nastaan XTAL2. Käytettäessä ulkoista, maksimissaan 18 MHz:n kellosignaalia, se liitetään nastaan XTAL1.

4.1.1 RC-oskillaattori

Mikrokontrolleri sisältää 6-bittisen TRIM-rekisterin, millä voidaan säätää RC-oskillaattorin taajuutta. Oskillaattorin taajuutta muutetaan ohjelmallisesti, mikä tallennetaan TRIM-rekisteriin. Mikäli mikrokontrolleri joudutaan resetoimaan, palautuu RC-oskillaattorin taajuudeksi tehdasasetuksena annettu 7,373 MHz ± 1 %.

4.1.2 Watchdog-oskillaattori

Mikrokontrollerin watchdog-lohko sisältää oman oskillaattorin, mikä värähtelee kiinteästi 400 kHz:n taajuudella. Kyseistä oskillaattoria käytetään erityisesti silloin, kun halutaan pienentää tehonkulutusta eikä tarvita korkeaa kellotaajuutta.

4.1.3 CCLK-järjestelmäkellon aktivointiviive

Mikrokontrollerissa on sisäänrakennettu aktivointiajastin, mikä viivästyttää järjestelmäkelloa hieman, kun mikrokontrolleri aktivoidaan. Kyseistä viivettä tarvitaan siksi, että järjestelmäkellon signaalilähde ehtii tasaantua. Käytettäessä ulkoista kidettä aktivointiviive on 992 OSCCLK-jaksoa ja lisäksi 60 µs – 100 µs. Mikäli käytetään mikrokontrollerin omaa RC-oskillaattoria, watchdog-oskillaattoria tai ulkoista signaalikelloa, viive on 224 OSCCLK-jaksoa ja lisäksi 60 µs – 100 µs.

4.1.4 DIVM-rekisteri

DIVM-rekisterin avulla voidaan muokata CCLK-järjestelmäkellon taajuutta. Operaatio toimii siten, että OSCCLK-signaali jaetaan jollakin positiivisella kokonaisluvulla. Näin saatu CCLK-signaalin taajuus voidaan laskea jopa 1/510-osaan alkuperäisestä signaalitaajuudesta. Kyseistä operaatiota käytetään silloin, kun halutaan vähentää tehonkulutusta eikä tarvita suurta toimintataajuutta. OSCCLK-signaalin jakaja voidaan muuttaa ohjelmallisesti.

4.2 Muistirakenne

P89LPC935-mikrokontrollerissa on yhteensä 768 tavua RAM-muistia, joka on jaettu kolmeen eri osaan. DATA-muistia on 128 tavua ja sitä voidaan osoittaa joko suoraan tai epäsuorasti. IDATA-muistia on 256 tavua ja sitä voidaan osoittaa vain epäsuorasti. IDATA-muistin osoiteavaruus sisältää myös DATA-muistin osoiteavaruuden. Lisäksi on 512 tavun suuruinen XDATA-muisti, jota käytetään tarvittaessa lisämuistina.

Yleisen käyttömuistin lisäksi mikrokontrollerissa on 512 tavua EEPROM-muistia, jota voidaan sekä lukea että kirjoittaa erityisen SFR-rekisterin kautta. EEPROM-muisti

sisältää erilaisia mikrokontrollerin tunnistustietoja ja joitakin tehdasasetuksia. Tämän lisäksi mikrokontrollerissa on vielä 16kB koodimuistia, jota käytetään osana ohjelman ajoa.

4.3 Keskeytykset

Käytettävä keskeytysrakenne jakautuu neljään eri prioriteettiin. P89LPC935-mikrokontrollerissa on seuraavat keskeytyslähteet:

- ulkoiset keskeytykset 0 ja 1
- ajastimet 0 ja 1
- sarjaportin lähetys
- sarjaportin vastaanotto
- sarjaportin yhdistetty lähetys/vastaanotto
- watchdog/reaaliaikakello
- I^2C-väylä
- näppäimistö
- komparaattorit 1 ja 2
- SPI (Serial Peripheral Interface)
- CCU (Capture/Compare Unit)
- EEPROM-muistin kirjoitus
- A/D-muunnoksen valmistuminen
- sähkökatkon tunnistin.

Jokainen keskeytyslähde voidaan erikseen sallia tai estää. Tämä tapahtuu joko asettamalla tai nollaamalla keskeytysrekisterien IEN0 ja IEN1 sisältö. IEN0-rekisteri sisältää lisäksi globaalin EA-bitin, millä kaikki keskeytykset voidaan estää.

Kaikkien keskeytyslähteiden prioriteettia voidaan muuttaa. Kyseinen operaatio tapahtuu muuttamalla prioriteettirekistereiden IP0, IP0H, IP1 ja IP1H bittisisältöä. Mikäli ohjelma palvelee tiettyä keskeytystä, sen voi sivuuttaa vain korkeamman prioriteetin keskeytyslähde. Kahden saman prioriteetin keskeytyslähteen keskinäinen palvelujärjestys määräytyy sisäisen äänestyssekvenssin perusteella.

4.4 I/O-portit

I/O-portteja on kaikkiaan neljä kappaletta: Port 0, Port 1, Port 2 ja Port 3. Kolme ensimmäistä porttia ovat 8-bittisiä ja Port 3 on 2-bittinen. Käytössä olevien I/O-nastojen määrä riippuu käytössä olevista kello- ja resetointiasetuksista. Käytettävissä olevien nastojen määrä vaihtelee 23 ja 26 nastan välillä.

Port 0, Port 2 ja Port 3 ovat ohjelmoitavissa joko tulo- tai lähtöporteiksi. Porttien jokainen bitti voidaan erikseen ohjelmoida. Ainoastaan Port 1:n bitit P1.4/INT1 ja P1.5/RST ovat aina tulonastoja eikä niitä voi vaihtaa edes ohjelmallisesti. Bittiä

P1.4/INT1 käytetään ulkoisen keskeytyslähteen tulona ja bittiä P1.5/RST ulkoisen reset-signaalin tulona.

4.4.1 Virtuaalinen kaksisuuntainen lähtö

Virtuaalista lähtöä voidaan käyttää sekä lähtönä että tulona ilman, että sitä tarvitsee konfiguroida uudelleen. Tämä on mahdollista siksi, että portin lähdön ollessa looginen ”1”, signaali on ohjattu niin heikosti, että ulkoinen laite voi vetää lähdön takaisin loogiselle 0-tasolle. Portin lähdön ollessa looginen ”0” nastan läpi voi kulkea suuriakin virtoja. Lisäksi lähdössä on kolme ylösvetotransistoria, joilla ohjaillaan signaalitasoja. Virtuaalinen kaksisuuntainen lähtö voidaan toteuttaa millä tahansa mikrokontrollerin neljästä I/O-portista.

P89LPC935-mikrokontrolleri toimii kolmen voltin jännitteellä, mutta nastat ovat yhteensopivia myös viiden voltin jännitteen kanssa. Viiden voltin jännitteen käyttö kuitenkin lisää mikrokontrollerin virrankulutusta, koska silloin I/O-nastasta vuotaa virtaa V_{dd}-nastaan.

4.4.2 Avokollektorilähtö

Mikäli avokollektorilähdön looginen taso on ”0”, käytetään ainoastaan piirin alasveto-transistoreita. Jos porttia käytetään loogisena ulostulona, tarvitaan ulkoista ylösvetoa. Yleensä tähän tarkoitukseen käytetään ylösvetovastusta, joka kytketään V_{dd}-nastaan. Kaikki mikrokontrollerin I/O-porttien nastat lukuunottamatta nastaa P1.5 (RST) voidaan valita ohjelmallisesti avokollektorilähdöksi.

4.4.3 Vuorovaihelähtö

Vuorovaihelähdön alasvetorakenne on samanlainen kuin avokollektori- ja virtuaalisella kaksisuuntaisella lähdöllä. Vuorovaihelähdöksi voidaan ohjelmoida kaikki muut I/O-porttien nastat, paitsi P1.5 (RST), P1.2 (SCL/T0) ja P1.3 (SDA/INT0).

4.4.4 Tuloportti

Mikä tahansa mikrokontrollerin neljästä I/O-portista voidaan asettaa ohjelmallisesti tuloportiksi. Tällöin portissa käytetään Schmitt-liipaisinta ja erillistä häiriönpoistopiiriä.

4.5 Käyttöjännitteen seuranta

Mikrokontrollerissa on erilliset käyttöjännitettä valvovat toiminnot, jotka pyrkivät takaamaan mikrokontrollerin oikean käyttäytymisen eri tilanteissa. Tällaisia tilanteita

ovat esimerkiksi käyttöjännitteen kytkeminen mikrokontrolleriin, sähkökatko ja käyttöjännitteen muuttuminen toiminnan aikana.

4.5.1 Sähkökatkon tunnistus

Mikrokontrollerissa on sisäänrakennettu piiri, joka seuraa käyttöjännitteen tasoa. Jännitteen pudotessa alle tietyn pisteen piirin ensisijainen tarkoitus on resetoida mikrokontrollerin prosessori. Piiri voidaan ohjelmoida myös siten, että se aiheuttaa ainoastaan keskeytyksen jännitteen pudotessa. Piirin toiminta voidaan joko sallia tai estää ohjelmallisesti.

Piirin ollessa aktivoituna mikrokontrollerin käyttöjännite voi vaihdella 2,7 V – 3,6 V. Piirin ollessa passiivisena vastaavat käyttöjännitteet ovat 2,4 V – 3,6 V. Käyttöjännitteen pudotessa alemmalle tasolle, voi seurauksena olla mikrokontrollerin väärä toiminta ja jossain vaiheessa myös toiminnan loppuminen kokonaan. Vastaavasti liian korkea käyttöjännite voi vaurioittaa piiriä pysyvästi.

4.5.2 Käyttöjännitteen kytkentä

Mikrokontrollerissa on erillinen funktio käyttöjännitteen kytkemisen seurannalla. Kyseistä funktiota tarvitaan silloin, kun käyttöjännite ei ole vielä saavuttanut tasoa, jolla sähkökatkon tunnistuspiiri voisi toimia. Kontrollerin RSTSRC-rekisteri sisältää POF-lipun, jonka asento kertoo käyttöjännitteen kytkemisen tilan. POF-lippu asettuu ylös, kun käyttöjännite kytketään, ja se voidaan nollata ohjelmallisesti.

4.6 Reset-signaalit

Mikrokontrollerissa on erillinen resetointinasta P1.5/RST, ja sitä voidaan käyttää myös normaalina digitaalisena sisäänmenona. Valinta tehdään vaihtamalla UCFG1-rekisterin bittiä RPE. Bitin ollessa tilassa ”1” nastaan voidaan syöttää ulkoinen reset-signaali. Muuten nasta toimii tavallisena piirin sisäänmenona.

Mikrokontrollerin resetointi voidaan toteuttaa seuraavilla lähteillä:

- ulkoisella resetointinastalla
- sähkökatkontunnistuksella
- watchdog-ajastimella
- ohjelmallisella resetoinnilla
- käyttöjännitte kytkemällä
- UART-keskeytyksellä.

Jokaiselle resetointilähteelle on oma bittinsä RSTSRC-rekisterissä. Näin voidaan tarkastella sitä, mikä lähde kulloinkin on aiheuttanut mikrokontrollerin resetoinnin.

Mikrokontrollerin resetoinnin jälkeen se hakee parametrit joko osoitteesta 0000H tai erillisestä käynnistysmuistista. Käynnistysmuistia käytetään silloin, kun UART on aiheuttanut resetoinnin, BOOTSTAT.0 -bitti on tilassa ”1” tai mikrokontrolleri on pakotettu ISP-tilaan käynnistyksen aikana. Muulloin parametrit haetaan osoitteesta 0000H.

4.7 Ajastimet ja laskurit

P89LPC935-mikrokontrolleri sisältää kaksi yleiseen käyttöön tarkoitettua ajastinta/laskuria: ajastin 0 ja ajastin 1. Molempia ajastimia voidaan käyttää myös laskureina. Ajastinkäytössä rekisterin sisältö kasvaa yhdellä jokaisella CCLK-signaalijaksolla. Laskurikäytössä rekisteri laskee bittimuutoksia ”1”→”0”. Ulkoisten laitteiden bittimuutokset ohjataan mikrokontrollerin nastoihin T0 ja T1. Kontrolleri rekisteröi tapahtumat nastoissa, mikäli ne kestävät vähintään yhden CCLK-signaalijakson verran.

Ajastimilla on viisi erilaista toimintatapaa (tapa 0, 1, 2, 3 ja 6.). Kaikki muut toimintatavat ovat yhtenevät molemmilla ajastimilla, paitsi tapa 3.

Toimintatapa 0:n ollessa käytössä käytetään 8-bittistä laskuria ja 32-esijakajaa. Vastaavasti ajastinrekisterit toimivat 13-bittisessä tilassa.

Toimintatapa 1 vastaa edellistä muilta osin, mutta silloin on käytössä kaikki ajastimen 16 bittiä.

Toimintatapa 2 konfiguroi ajastinrekisterit 8-bittisiksi laskureiksi, jotka sisältävät lisäksi automaattisen uudelleenlatauksen.

Toimintatapa 3 pysäyttää ajastimen 1:n. Sitä voidaan kuitenkin käyttää sarjaliikenneportin baudinopeusgeneraattorina. Ajastin 0 puolestaan toimii kahtena erillisenä 8-bittisenä ajastimena.

Toimintatapa 6:ssa ajastinta käytetään pulssinleveysmodulaattorikäytössä (PWM).

Molemmissa ajastimissa on ylivuotoulostulo, joka voidaan halutessa aktivoida ylivuoden sattuessa. Ulostulonastoina käytetään nastoja T0 ja T1.

4.8 RTC

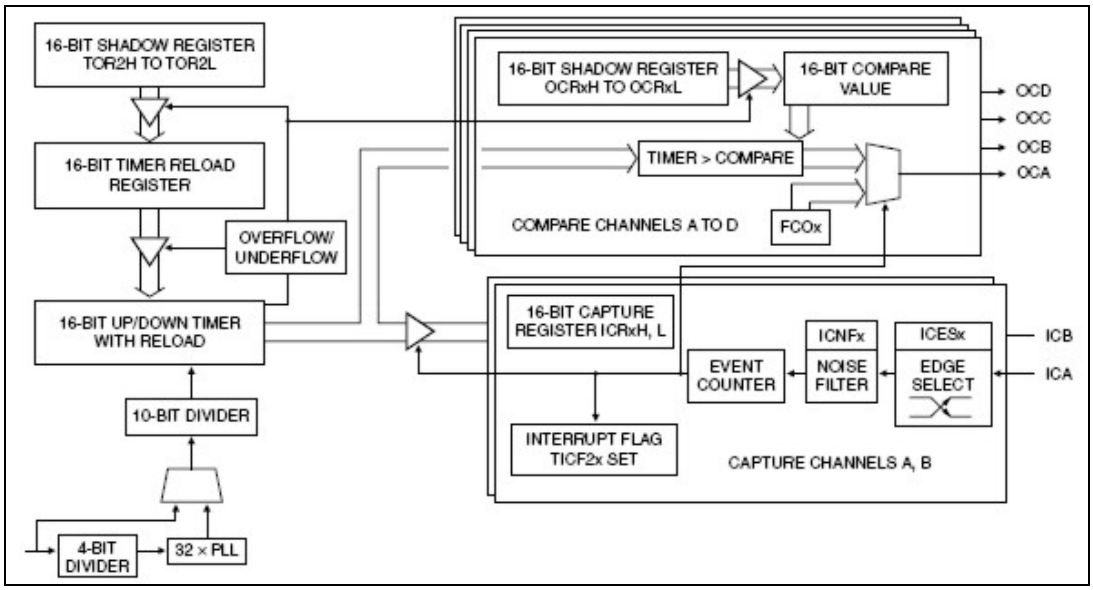
Mikrokontrollerissa on yksinkertainen sisäänrakennettu reaaliaikakello (RTC), jota voidaan käyttää myös silloin, kun muu laite on kytketty pois päältä. Reaaliaikakelloa voidaan käyttää muun laitteen aktivoimiseen tai erillisenä keskeytyslähteenä. RTC on rakenteeltaan 23-bittinen johon sisältyy 7-bittinen esijakaja. Laskuri on alaslaskuri, ja sen saavuttaessa arvon ”0” se ladataan uudelleen ja asetetaan RTCF-lippu ylös.

RTC:n kellosignaalina voidaan käyttää joko CCLK-signaalia tai XTAL-oskillaattoria. CCLK-signaalia käytetään silloin, kun XTAL-oskillaattori on CPU-kellon käytössä. Reaaliaikakello resetoituu vain silloin, kun laitteeseen kytketään käyttöjännite.

4.9 CCU-yksikkö

CCU-yksikön (Capture/Compare Unit) capture-lohkon tehtävä on seurata siihen liitettyjä tulosignaaleja ja signaalimuutoksen tapahtuessa tallettaa oman 16-bittisen laskurinsa senhetkinen arvo. Tallennus voidaan määrittää ohjelmallisesti joko nousevalle tai laskevalle signaalireunalle. Kyseisellä toiminnolla pystytään mittaamaan esimerkiksi jonkin tapahtuman tai suorituksen ajallista kestoa. Kuvasta 3 nähdään capture-lohkon tulosignaalien nastat ICA ja ICB.

Compare-lohkon tehtävä on verrata laskurin nykyistä arvoa aiemmin tietyllä ajanhetkellä talletettuun arvoon. Kun laskurin arvo vastaa CCU-yksikön ohjausrekisteriin tallennettua arvoa, keskeytyslippu asettuu. Tästä aiheutuu keskeytys, mikäli se on ohjelmallisesti sallittu. Keskeytyksen sattuessa kuvassa 5 näkyvän compare-lohkon lähtöjen OCA, OCB, OCC ja OCD tilaa voidaan vaihtaa. Näin pystytään generoimaan pulssisuhteeltaan vaihtelevaa kanttiaaltoa.



Kuva 3. CCU-yksikön lohkokaavio [5]

4.9.1 CCU-kello

CCU-yksikkö käyttää kellosignaalina joko PCLK-signaalia (Peripheral Clock) tai PLL-ulostuloa (Phase-locked Loop). Vaihelukittu silmukka on suunniteltu toimimaan 0,5 MHz – 1,0 MHz:n oskillaattorilla. Signaali kerrotaan edelleen 32:lla, jolloin saadaan pulssinmuokkauksessa käytettävä 16 MHz – 32 MHz:n CCUCLK-signaali. Lisäksi

vaihelukittu silmukka sisältää 4-bittisen jakajan, jota käytetään PCLK-signaalin jakamiseen.

4.9.2 Kellosignaalin esijakaja

CCU-yksikkö sisältää 10-bittisen juoksevan laskurin, jolla voidaan jakaa CCUCLK-signaalia. Laskuri voidaan ohjelmoida lataamaan itsensä uudelleen ylivuodon sattuessa.

4.9.3 Ajastinpiiri

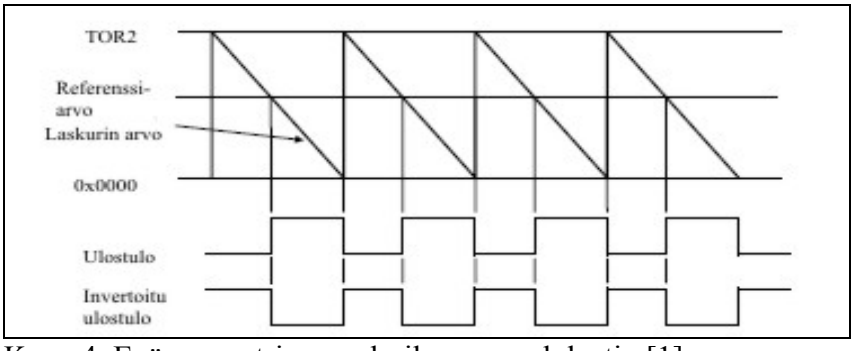
CCU-yksikön käyttämä ajastinpiiri on vapaasti juokseva 16-bittinen laskuri, jonka suunta voidaan määrätä erillisellä ohjausbitillä. Laskurin suunta voidaan kääntää myös sen ollessa toiminnassa. Tällöin laskuri jatkaa senhetkisestä arvostaan asetettuun suuntaan. Laskuria voidaan lukea tai siihen voidaan kirjoittaa milloin vain. Laskuria voidaan käyttää myös vastaavana 8-bittisenä laskurina.
Laskurin uudelleenlatauksen yhteydessä CCU-yksikön laskurin ylivuodon osoittama keskeytyslippu asettuu ja aiheuttaa keskeytyksen, mikäli se on ohjelmallisesti sallittu.

4.9.4 Kohinasuodatin

CCU-yksikkö sisältää yksinkertaisen kohinasuodattimen, jota käytetään capture-lohkon yhteydessä. Suodatin saadaan käyttöön asettamalla CNF-bitti (Capture Noise Filter). Suodattimelle täytyy syöttää alussa hetki samaa signaalia, joko loogista ”1”:a tai loogista”0”:a, jotta se oppii tunnistamaan signaalireunan. Kohinasuodatin tarvitsee neljä signaalinäytettä, jotka se ottaa joka toisella CCLK-jaksolla.

4.9.5 Pulssinleveysmodulaattori

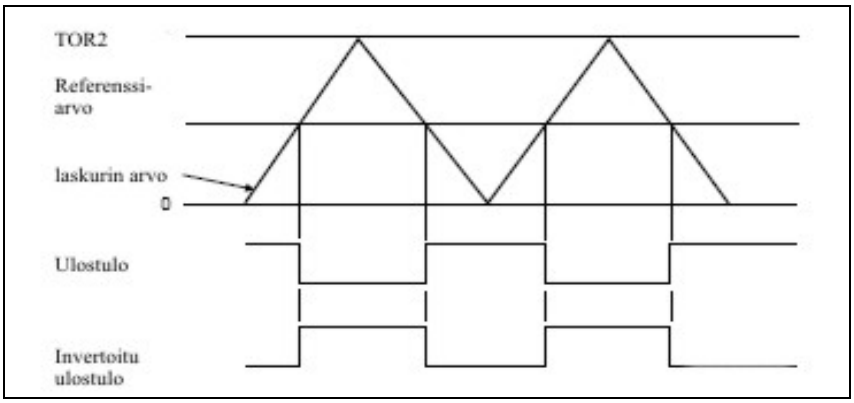
CCU-yksikön pulssinleveysmodulaatio jakautuu symmetriseen ja epäsymmetriseen. Epäsymmetristä pulssinmuokkausta käytettäessä CCU-yksikön laskuri toimii ainoastaan laskevasti, riippumatta erillisestä suunnanohjausbitistä.



Kuva 4. Epäsymmetrinen pulssileveysmodulaatio [1]

Kuvasta 4 nähdään epäsymmetrisen pulssileveysmodulaation periaate. Alussa laskuri ladataan TOR2-rekisterin (Time-Out rekister) arvolla. Tämän jälkeen laskurin arvo vähenee lineaarisesti nollaan asti. Seuraavaksi laskuri ladataan uudelleen ja operaatio alkaa alusta. Referenssiarvolla voidaan muuttaa ulostulevan signaalin pulssisuhdetta. Mitä suurempi referenssiarvo on, sitä suurempaa pulssisuhdetta voidaan generoida. Ulostulon lisäksi signaali voidaan kääntää, jolloin saadaan invertoitu ulostulo.

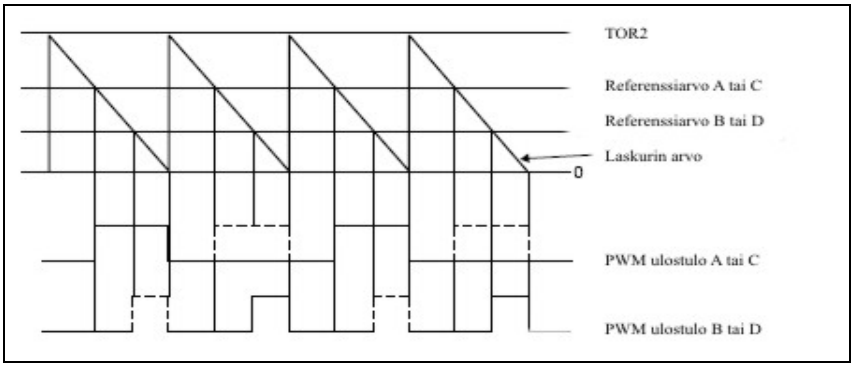
Symmetrisessä pulssileveysmodulaatiossa CCU-yksikön laskuri toimii vuorotellen laskevana ja nousevana laskurina.



Kuva 5. Symmetrinen pulssileveysmodulaatio [1]

Kuvasta 5 nähdään symmetrisen pulssileveysmodulaation periaate. Kuten epäsymmetrisessä pulssileveysmodulaatiossa, laskuria ei alussa ladata tietyllä arvolla, vaan laskurin arvo lähtee kasvavaan nollasta. Arvo kasvaa aina TOR2-rekisterissä olevaan arvoon asti, minkä jälkeen laskurin arvo lähtee vähenemään. Arvoa vähennetään niin kauan, kunnes saavutetaan arvo 0. Tämän jälkeen operaatio toistetaan niin monta kertaa kuin on tarpeen. Referenssiarvolla ohjataan pulssisuhdetta aivan kuten epäsymmetrisessä pulssileveysmodulaatiossakin.

Edellä esiteltyjen pulssileveysmodulaatioiden lisäksi voidaan käyttää vielä niin sanottua vaihtelevan ulostulon pulssileveysmodulaatiota. Kyseistä toimintatilaa käytettäessä pulssileveysmodulaattorin täytyy olla epäsymmetrisessä tilassa. Kanavat A/B ja C/D voidaan valita ohjelmallisesti pareiksi, joilla ohjataan pulssisuhdetta vuoronperään.



Kuva 6. Vaihtelevan ulostulon pulssileveysmodulaatio [1]

Kuvasta 6 nähdään vaihtelevan ulostulon pulssileveysmodulaation toimintaperiaate. Operaatio on laskurin osalta täysin samanlainen, kuin normaalissa epäsymmetrisessä pulssileveysmodulaatiossa. Operaatiot eroavat toisistaan siinä, että vaihtelevan ulostulon pulssileveysmodulaatiossa voidaan käyttää jopa neljää referenssiarvoa, millä pulssisuhdetta ohjataan.

Esimerkkinä käytettäköön referenssiarvoja A ja B. Ensimmäisellä laskurin laskentakierroksella arvoa verrataan referenssiarvoon A. Toisella laskentakierroksella käytettävä referenssiarvo on B. Referenssiarvoa vaihdetaan jokaisella laskentakierroksella, minkä vuoksi ulostulot A ja B ovat aktiivisina vain joka toisella syklillä. Kuvassa 6 on merkitty katkoviivoilla ulostulosta pois jäävät syklit.

4.10 UART

Lyhenne UART tulee sanoista Universal Asynchronous Receiver/Transmitter, mikä tarkoittaa yleistä asynkronista vastaanotinta/lähetintä. UART on toisin sanoen hyvin yleisesti käytetty sarjaliikennepiiri, jolla voidaan muuttaa rinnakkaismuotoista dataa sarjamuotoiseksi, ja päinvastoin.

P89LPC935-mikrokontrollerin UART-piirin siirtonopeus voidaan generoida kolmesta eri lähteestä: oskillaattorilta, ajastin 1:n ylivuodosta tai erillisestä siirtonopeusgeneraattorista. UART-piirin perusominaisuuksiin kuuluu tahdistusvirheiden huomaaminen, automaattinen osoitteentunnistaminen, valittavissa oleva kaksoispuskurointi sekä useita keskeytysvaihtoehtoja. UART voi toimia neljässä eri moodissa.

4.10.1 Moodi 0

Moodi 0:ssa sarjamuotoinen data sekä lähetetään että vastaanotetaan RXD-linjassa.. Data liikkuu 8-bitin merkkeinä siten, että vähiten merkitsevä bitti (LSB) tulee ensin. Käytettävä siirtonopeus on $^1/_{16}$ CPU:n kellotaajuudesta. Moodi 0:ssa tiedonsiirto tapahtuu synkronisesti ja sen siirtokellona toimii TXD-linja.

4.10.2 Moodi 1

Moodi 1:ssä data lähetetään TXD-linjaa pitkin ja vastaanotto tapahtuu RXD-linjassa. Liikkuvien merkkien koko on 10 bittiä. Ensimmäisenä on alkubitti ”0”, sen jälkeen 8 databittiä LSB ensin ja lopuksi loppubitti ”1”. Kun data on vastaanotettu, loppubitti tallennetaan RB8-bittiin erikoisrekisterin SCON-osaan (Serial Port Control). siirtonopeus vaihtelee sen mukaan, käytetäänkö ajastin 1:tä vai erillistä siirtonopeusgeneraattoria lähteenä. Moodi 1:ssä tiedonsiirto on asynkronista.

4.10.3 Moodi 2

Myös moodi 2:ssa datan lähetys hoidetaan TXD-linjaa pitkin ja vastaavasti datan vastaanotto RXD-linjaa pitkin. Merkkien koko on kyseisessä moodissa 11 bittiä. Paketissa ensimmäisenä on alkubitti ”0” ja sen jälkeen tulee 8 databittiä LSB ensin. Tämän jälkeen tulee ohjelmoitava yhdeksäs databitti ja lopuksi vielä loppubitti ”1”.

Datan lähetyksen jälkeen yhdeksännelle databitille voidaan määrätä arvo ”0” tai ”1” ja tallettaa se SCON-rekisterin TB8-bittiin. Vaihtoehtoisesti myös PSW:n (Program Status Word) pariteettibitti P voidaan tallettaa TB8-bittiin.

Datan vastaanoton jälkeen yhdeksännen databitin arvo talletetaan SCON-rekisteriin RB8-bitin sisällöksi, eikä loppubittiä tallenneta. Siirtonopeus voidaan ohjelmoita joko $^1/_{16}$ tai $^1/_{32}$ osaan CPU:n kellotaajuudesta.

4.10.4 Moodi 3

Moodi 3 on merkkien koon ja tiedonsiirtokanavien suhteen identtinen moodi 2:n kanssa. Ainoa ero on siirtonopeus. Moodi 3:n siirtonopeus vaihtelee ja se määräytyy joko ajastin 1:n ylivuotonopeudesta tai erillisestä siirtonopeusgeneraattorista.

4.10.5 Siirtonopeusgeneraattori

Mikrokontrollerin UART-piiri sisältää erillisen siirtonopeusgeneraattorin, jolla siirtonopeutta voidaan säädellä. Siirtonopeuden määrittäminen perustuu BRGR1:een (Baud Rate Generator Register) ja BRGR0:aan tallennettuihin esiohjelmoituihin arvoihin. Kyseiset rekisterit muodostavat 16-bittisen jakaja-arvon, jolla siirtonopeutta ohjataan. Erillistä siirtonopeusgeneraattoria käyttämällä saadaan siirtonopeutta ohjattua tarkemmin kuin käyttämällä ajastin 1:tä.

4.10.6 Tahdistusvirhe

Tiedonsiirrossa tapahtuu tahdistusvirhe silloin, jos loppubitti havaitaan ”0”:ksi. Siitä viedään tieto erilliseen tilarekisteriin (SSTAT). Mikäli tehonvalvontarekisterin (PCON) SMOD0-bitin sisältö on ”1”, tahdistusvirheet voidaan tallentaa SCON-rekisterin bittiin SCON.7. Jos SMOD0-bitin arvo on ”0”, SCON.7-bitin sisältö on sama kuin SM0:n.

4.10.7 Break-tilan tunnistus

Break-tilasta sarjaliikenneyksikkö tallentaa tiedon SSTAT-rekisteriin aivan kuten tahdistusvirheenkin sattuessa. Break-tila tunnistetaan silloin, kun havaitaan 11

peräkkäistä bittiä nollaksi. Break-tilan tunnistamista käytetään yleensä laitteen uudelleenkäynnistämiseen tai sen pakottamiseen ISP-tilaan.

4.10.8 Kaksoispuskurointi

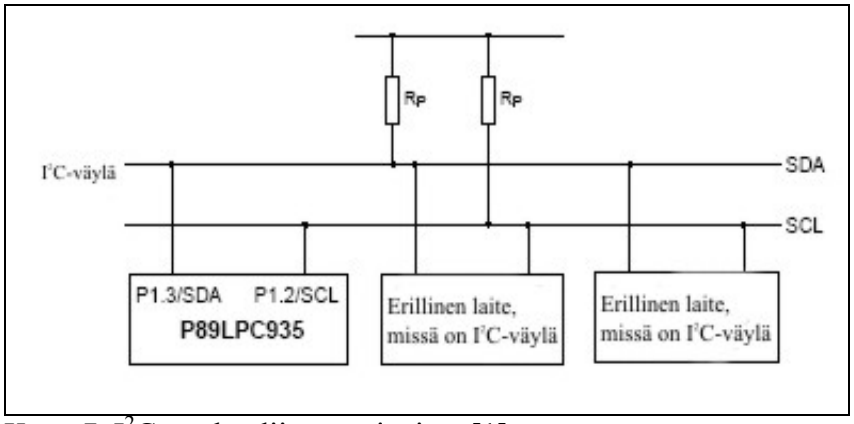
UART-piiri käyttää tiedonlähetyksessä kaksoispuskurointia. Tämä mahdollistaa seuraavan merkin kirjoittamisen SBUF-rekisteriin (Serial Port Data Buffer register) odottamaan, kun edellistä merkkiä lähetetään. Kaksoispuskuroinnin avulla voidaan lähettää merkkijono käyttämällä vain yhtä lopetusbittiä merkkien välissä.

Kaksoispuskurointi voidaan estää ohjelmallisesti. Tällöin SSTAT.7-bitti saa arvon ”0”. Kaksoispuskurointia voidaan käyttää vain silloin, kun datanlähetys tapahtuu eri linjaa pitkin kuin datan vastaanotto. Näin ollen moodi 0:a käytettäessä kaksoispuskurointi täytyy estää. Kaksoispuskurointia käytettäessä lähetyskeskeytys generoituu silloin, kun SBUF-rekisteri on valmis vastaanottamaan uutta dataa.

4.11 I^2C-väylä

Mikrokontrollerin I^2C-väylää käytetään ulkoisten laitteiden liittämisessä mikrokontrolleriin. Väylä muodostuu kahdesta johdosta, SDA ja SCL. SDA-johtoa käytetään datan siirtämiseen laitteiden välillä ja vastaavasti SCL-johto on kellosignaalia varten. I^2C-väylän ominaisuuksia ovat muun muassa:

- Väylää voidaan käyttää kaksisuuntaisessa tiedonsiirrossa isännän ja orjan välillä.
- Väylässä voidaan käyttää useaa isäntälaitetta.
- Kellosignaalin synkronointi mahdollistaa kahden eri nopeuksisen laitteen kommunikoinnin samassa väylässä.
- Kellosignaalin synkronointia voidaan käyttää kättelymekanismina tiedonsiirrossa.
- Väylää voidaan käyttää testi- ja diagnostiikkasovelluksiin.



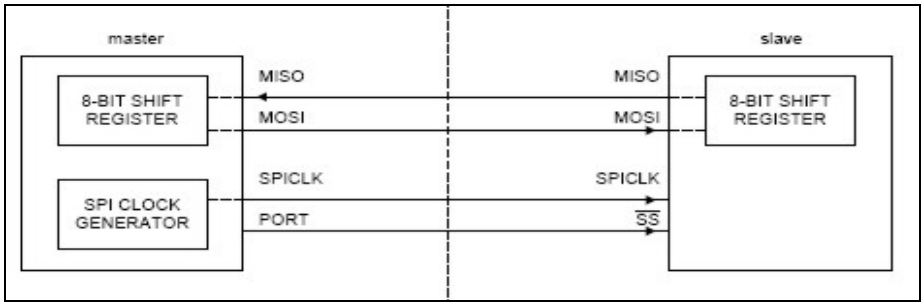
Kuva 7. I^2C-väylän liityntärajapinta [1]

Kuvassa 7 on tyypillinen I^2C-väylän liityntärajapinta. Väylään on kiinnittyneenä P89LPC935-mikrokontrollerin lisäksi kaksi ulkoista laitetta, jotka käyttävät I^2C-protokollaa.

4.12 SPI

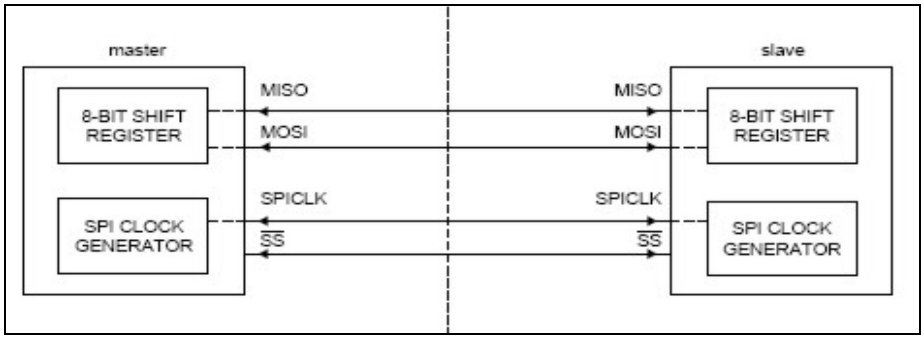
P89LPC935-mikrokontrollerissa on myös toinen sarjaliikenneväylä SPI, joka on tarkoitettu ulkoisten I/O-laitteiden liittämiseen. Se on nopea synkroninen kokodupleksi väylä, ja lisäksi se voi toimia kahdessa eri tilassa. Isäntänä se toimii maksimissaan 3 Mbit/s nopeudella ja orjana vastaavasti maksimissaan 2 Mbit/s nopeutta.

SPI-rajapinta koostuu neljästä erillisestä signaalista: SPICLK, MOSI, MISO ja SS. Kolmea ensin mainittua signaalia käytetään erillisten SPI-rajapintaa tukevien laitteiden liittämisessä toisiinsa. Kellosignaalia (SPICLK) käytetään isäntätilassa ulostulona ja vastaavasti orjatilassa sisäänmenona. MOSI-signaalia käytetään silloin, kun data liikkuu isäntälaitteesta orjalaitteeseen. Vastaavasti MISO-signaalia käytetään silloin, kun datan kulkusuunta on orjalaitteesta isäntälaitteeseen. Isäntälaite valitsee aina yhden laitteen senhetkiseksi orjalaitteeksi. Valittu laite käyttää SS-signaalia, jolla se ilmoittaa olevansa orjalaite.



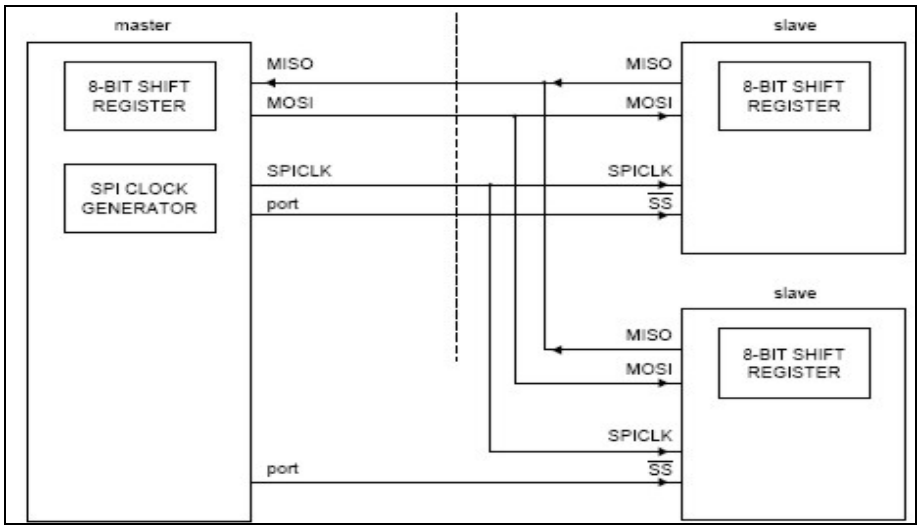
Kuva 8. SPI-kokoonpano, jossa on yksi isäntälaite ja yksi orjalaite [1]

Kuvassa 8 on esitettynä tyypillinen SPI-väylä, jossa on yksi isäntälaite ja yksi orjalaite. Kuvassa olevat nuolet kertovat datan kulkusuunnan kyseisessä linjassa. Vasemmalla puolella on isäntälaite ja vastaavasti oikealla puolella orjalaite.



Kuva 9. SPI-kokoonpano, jossa molemmat laitteet voivat toimia joko isäntänä tai orjana [1]

Kuvassa 9 on SPI-kokoonpano, jossa molemmat laitteet voivat toimia joko isäntänä tai orjana. Valinta suoritetaan SS-linjassa kulkevalla signaalilla. Laite, jonka SS-porttiin lähetetään ensin signaali, toimii senhetkisenä orjalaitteena.

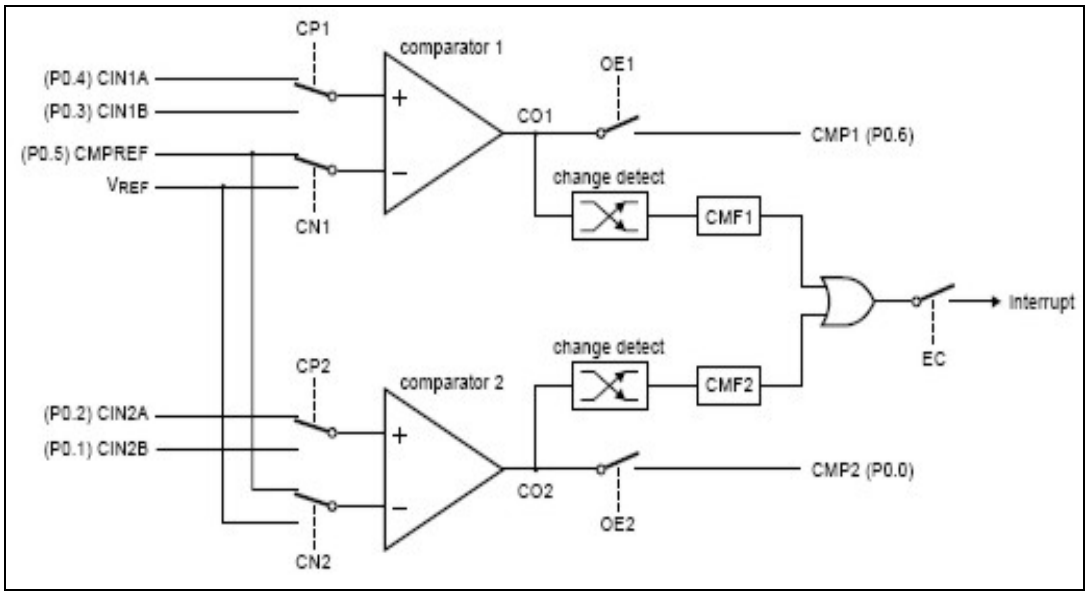


Kuva 10. SPI-kokoonpano, jossa on yksi isäntälaite ja useampia orjalaitteita [1]

10 9 nähdään SPI-kokoonpano, joka muodostuu yhdestä isäntälaitteesta ja useammasta orjalaitteesta. Toimintaperiaate on identtinen aikaisempien kokoonpanojen kanssa. Kaikki orjalaitteet käyttävät SS-nastaa, johon isäntälaite lähettää signaalin omasta portistaan.

4.13 Analogiset komparaattorit

P89LPC935-mikrokontrolleri on varustettu kahdella analogisella komparaattorilla. Komparaattorien asetuksia voidaan muokata monipuolisesti, mutta perustoimintaperiaate on yksinkertainen. Komparaattorissa on kaksi sisäänmenoa: positiivinen ja negatiivinen. Mikäli positiivinen sisäänmeno on negatiivista suurempi, komparaattorin ulostulo on ”1”. Muissa tapauksissa ulostulo on ”0”. Ulostulo voidaan tallettaa rekisteriin ja/tai ohjata erilliseen nastaan. Lisäksi komparaattorit voidaan ohjelmoida aiheuttamaan keskeytyksen, mikäli ulostulo muuttuu.



Kuva 11. Analogisten komparaattorien kytkentäkaavio [1]

Kuvassa 11 on komparaattorien nastojen kytkeminen. Komparaattorien syötettävät signaalit kytketään nastoihin CIN1A, CIN1B, CIN2A tai CIN2B. Tulosignaalit voidaan valita vielä erillisillä kytkimillä CP1 ja CP2. Vertailusignaali kytketään nastaan CMPREF, tai mikäli halutaan käyttää mikrokontrollerin sisäistä 1,23 V ± 10 % referenssijännitettä, nastaan V_{ref}.

Komparaattorien ulostulot johdetaan lohkoon, joka tunnistaa niissä tapahtuvat muutokset. Jokainen tilamuutos aiheuttaa CMF1-tai CMF2-rekisterin lippubitin asettumisen. Keskeytys voidaan sallia ohjelmallisesti erillisen kytkimen EC-avulla tilamuutoksen sattuessa. Komparaattorien ulostulot voidaan lisäksi ohjata nastoihin CPM1 ja CMP2 erillisten kytkimien OE1 ja OE2 avulla.

4.14 KBI-keskeytys

KBI-keskeytyksen (Keypad Interrupt) generointiin käytetään port 0:an tulevaa signaalia. Sen arvoa verrataan ennalta määrättyyn referenssiarvoon. Portti voidaan ohjelmoida SFR:n avulla eri käyttötarkoituksia varten. Keskeytyslähteenä voidaan käyttää esimerkiksi näppäimistöä tai väyläosoitteen tunnistusta.

Näppäimistökeskeytyksen maskirekisteri (KBMASK) valitsee portti 0: n yhdistetyistä sisääntulevista signaaleista ne, jotka voivat aiheuttaa keskeytyksen. Näppäimistökeskeytyksen rakennerekisteri (KBPATN) puolestaan valitsee sisällön, jota verrataan portti 0:n arvoon. Tämän lisäksi näppäimistökeskeytyksen ohjausrekisterissä (KBCON) oleva lippubitti (KBIF) asettuu silloin, kun verrattavat arvot ovat samat. Lippubitin asettuessa generoidaan keskeytys, mikäli se on erikseen käyttäjätasolla sallittu.

4.15 Watchdog-ajastin

Watchdog-ajastimen tarkoitus on seurata mikrokontrollerin ohjelmansuoritusta ja resetoida mikrokontrolleri, mikäli ohjelma jostain syystä sekoaa. Näin kaatunut ohjelmansuoritus voidaan aloittaa alusta ilman, että käyttäjän tarvitsee tehdä mitään.

Watchdog-ajastimen alivuoto aiheuttaa mikrokontrollerin resetoinnin. Alivuoto tapahtuu silloin, kun laskuri saavuttaa arvon ”0”, jollei sille ehditä syöttää uutta arvoa ennen sitä. Uusi arvo syötetään aina, kun ohjelma on ajettu onnistuneesti läpi.

P89LPC935-mikrokontrollerin watchdog-ajastimessa on 12-bittinen esijakaja ja 8-bittinen alaslaskin. Laskurin arvoa vähennetään esijakajalta saatavalla pulssilla. Esijakajan kellosignaalina käytetään joko PCLK/32-signaalia tai watchdog-ajastimen omaa 400 kHz:n oskillaattoria. Mikäli watchdog-ajastin ei ole käytössä, oskillaattoria voidaan käyttää sisäisenä ajastimena ja sillä voidaan generoida keskeytyksiä.

5 A/D–MUUNTIMET [1]

A/D-muuntimen tarkoitus on muuttaa analogisia signaaleita digitaalisiksi ja päinvastoin. Kyseisiä operaatioita tarvitaan esimerkiksi silloin, kun mikrokontrolleriin on liitetty ulkopuolisia antureita, jotka ovat yleensä analogisia laitteita.

P89LPC935-mikrokontrolleri sisältää kaksi 8-bittistä ja nelikanavaista multipleksattua A/D-muunninta, jotka käyttävät yhteistä ohjauslogiikkaa. Muita A/D-muunninyksikön ominaisuuksia ovat:

- neljä tulosrekisteriä kullekin muuntimelle
- kuusi toimintatapaa
- neljä eri tapaa aloittaa muunto-operaatio
- raja-arvokeskeytys
- kellosignaalin jakaja
- virransäästötila
- keskeytyksen generointi.

5.1 A/D-muuntimien toimintatavat

P89LPC935-mikrokontrollerin A/D-muuntimilla on kuusi eri toimintatapaa: kiinteän kanavan yksittäismuunnos, kiinteän kanavan jatkuva muunnos, automaattinen pyyhkäisy yksittäisellä muunnolla, automaattinen pyyhkäisy jatkuvalla muunnolla, kahden kanavan jatkuva muunnos sekä askeltoiminto.

5.1.1 Kiinteän kanavan yksittäismuunnos

Kiinteän kanavan yksittäismuunnossa voidaan valita yksi tulokanava, jonka signaali muunnetaan. Muunnos suoritetaan vain kerran ja tulos viedään kyseisen tulokanavan tulosrekisteriin. Mikäli keskeytykset on erikseen sallittu, se generoidaan muunnoksen suorittamisen jälkeen.

5.1.2 Kiinteän kanavan jatkuva muunnos

Kiinteän kanavan jatkuvassa muunnossa valitaan myös yksi tulokanava. Muunnoksia suoritetaan kuitenkin jatkuvalla syklillä ja muunnostulokset viedään neljään erilliseen tulosrekisteriin. Kun kaikissa neljässä tulosrekisterissä on muunnostulos, generoidaan keskeytys, mikäli se on ohjelmallisesti sallittu. Muunnoksia suoritetaan niin kauan, kunnes ne lopetetaan ohjelmallisesti. Uudet muunnostulokset viedään aina vanhojen tulosten tilalle tulosrekistereihin.

5.1.3 Automaattinen pyyhkäisy yksittäisellä muunnolla

Automaattisessa pyyhkäisyssä voidaan käyttää mitä tahansa neljän tulokanavan kombinaatiota. Jokaisesta valitusta tulosta tehdään muunnos, joka viedään tulosrekisteriin. Muunnos suoritetaan kerran, minkä jälkeen generoidaan keskeytys, mikäli se on sallittu. Mikäli valittuna on vain yksi tulokanava, kyseinen toimintatapa vastaa kiinteän kanavan yksittäismuunnosta.

5.1.4 Automaattinen pyyhkäisy jatkuvalla muunnolla

Kyseinen toimintatapa vastaa automaattista pyyhkäisyä yksittäisellä muunnolla, mutta erona on se, että muunnoksia tehdään jatkuvassa syklissä. Kaikkien tulokanavien muunnostulokset viedään niitä vastaaviin tulosrekistereihin. Tämän jälkeen generoidaan keskeytys, mikäli se on sallittu. Muunnos aloitetaan alusta siten, että ensin valittu tulokanava muunnetaan ensin ja prosessia toistetaan niin kauan, kunnes se lopetetaan ohjelmallisesti. Uudet muunnostulokset viedään tulosrekistereihin vanhojen tulosten tilalle.

5.1.5 Kahden kanavan jatkuva muunnos

Kahden kanavan jatkuvassa muunnoksessa voidaan valita kaksi tulokanavaa, joista muunnos suoritetaan. Molempien kanavien signaalimuunnos suoritetaan kaksi kertaa vuoronperään. Muunnostulokset talletetaan muunnosrekistereihin. Tämän jälkeen generoidaan keskeytys, mikäli se on sallittu. Sykliä toistetaan, kunnes muunnos lopetetaan ohjelmallisesti.

5.1.6 Askeltoiminto

Askeltoiminto mahdollistaa yksittäismuunnoksen automaattisessa signaalipyyhkäisyssä. Mitkä tahansa tulokanavat, joista muunnokset suoritetaan, voidaan valita ohjelmallisesti. Jokaisen yksittäisen muunnoksen jälkeen generoidaan keskeytys, mikäli se on sallittu. Tämän jälkeen A/D-yksikkö jää odottamaan seuraavaa signaalimuunnosta.

5.2 A/D-muunnoksen aloitustavat

A/D-muunnoksen aloittaminen voidaan valita neljästä eri vaihtoehdosta: ajastinliipaisu, välitön muunnoksen aloittaminen, signaalireunan liipaisu tai kaksoiskanavan välitön muunnoksen aloittaminen.

5.2.1 Ajastinliipaisu

A/D-muunnos aloitetaan, kun ajastin 0:ssa tapahtuu ylivuoto. Muunnosta tehtäessä kaikki muut ajastin 0:a koskevat liipaisut jätetään huomiotta, kunnes muunnos on valmis. Ajastinliipaisua voidaan käyttää kaikissa A/D-muuntimen toimintatavoissa.

5.2.2 Välitön muunnoksenaloitus

Kyseistä muunnoksen aloitustapaa käytettäessä signaalimuunnos aloitetaan välittömästi. Välitöntä signaalimuunnoksen aloitusta voidaan käyttää niin ikään kaikissa A/D-muuntimen toimintatavoissa.

5.2.3 Signaalireunan liipaisu

A/D-muunnos voidaan aloittaa myös signaalireunan muutoksesta. Signaali, jota halutaan tarkkailla, liitetään nastaan P1.4. Muunnoksen ollessa käynnissä muut signaalireunan liipaisut eivät ole käytössä. Signaalireunan liipaisua voidaan käyttää kaikissa A/D-muuntimen toimintatavoissa.

5.2.4 Kaksoismuunnoksen välitön muunnoksenaloitus

Kaksoismuunnoksessa molemmat A/D-muuntimet aloittavat signaalimuunnoksen synkronisesti. A/D-muuntimien toimintatapa voi olla mikä tahansa edellä esitetyistä, kunhan molemmat muuntimet ovat samassa toimintatilassa. Jatkuvassa signaalinmuunnoksessa molemmilla muuntimilla täytyy olla valittuna yhtä monta sisäänmenokanavaa. Signaalimuunnoksen alkaessa molemmat A/D-muuntimet aloittavat

muunnosoperaation samanaikaisesti, vaikka vain toinen muunnin olisi laukaissut muunnostapahtuman.

5.3 Raja-arvokeskeytys

Molemmat A/D-muuntimet sisältävät sekä alaraja- että ylärajarekisterin. Kun neljä MSB:tä on muunnettu, saatua arvoa verrataan raja-arvorekistereissä oleviin arvoihin. Mikäli muunnostulos ei osu raja-arvojen väliin, generoidaan keskeytys, jos se on sallittu. Jos muunnostulos osuu raja-arvojen väliin, muunnetaan loput neljä bittiä ja saatua 8-bittistä kokonaismuunnosta verrataan uudelleen raja-arvoihin. Mikäli muunnostulos ei osu raja-arvoihin, generoidaan keskeytys. Raja-arvokeskeytys voidaan joko sallia tai estää ohjelmallisesti.

5.4 Kellosignaalin jakaja

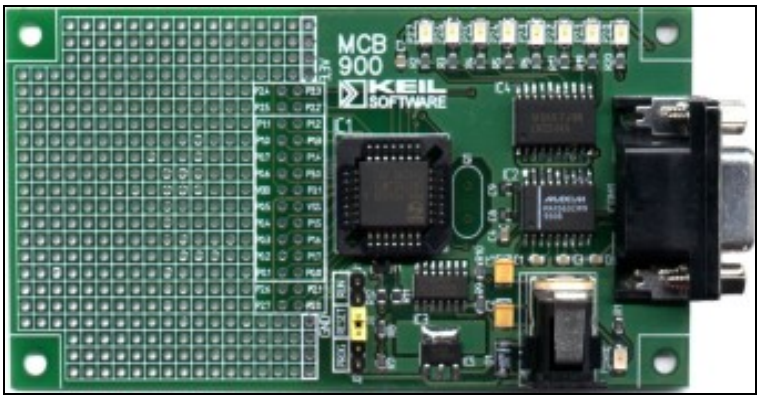
A/D-muunnin tarvitsee toimiakseen sisäisen kellosignaalin 500 kHz – 3,3 MHz. Tämän vuoksi A/D-muuntimessa on sisäänrakennettu kellosignaalin jakaja, joka voidaan ohjelmoida jakamaan muuntimelle tuleva kellosignaali kokonaisluvuilla 1 – 8. Jakajana voidaan käyttää myös parittomia lukuja, sillä rekisteri on 3-bittinen.

5.5 A/D-muuntimen sulkeminen ja tyhjäkäyntitila

Mikrokontrollerin ollessa tyhjäkäyntitilassa A/D-muunnin voi suorittaa signaalimuunnoksia. Muunnoksen valmistuttua A/D-muunnin herättää mikrokontrollerin toimintatilaan, mikäli se on erikseen sallittu. A/D-muuntimen ollessa aktiivisena se kuluttaa aina virtaa, vaikkei se suorittaisikaan signaalimuunnoksia. Näin virrankulutuksen minimoimiseksi A/D-muunnin kannattaa kytkeä pois päältä.

6 MCB900-LAITEALUSTA JA TESTIOHJELMAT

P89LPC935-mikrokontrolleri on yhdistetty Keilin MCB900-testialustaan mikrokontrollerin testaamisen helpottamiseksi. Testialustassa on RS232-liityntä, jolla se voidaan yhdistää tietokoneen COM-porttiin sarjakaapelin avulla. Lisäksi pakettiin kuuluu µVision-ohjelmisto, jolla luodaan yhteys testialustan ja tietokoneen välille. Ohjelmistolla voidaan myös ladata ajettavat ohjelmat mikrokontrolleriin, seurata ohjelmansuoritusta ja rekisterisisältöjä sekä ajaa uusia parametrejä mikrokontrollerin flash-muistiin. Kuvassa 12 on Keil MCB900 -testialusta.

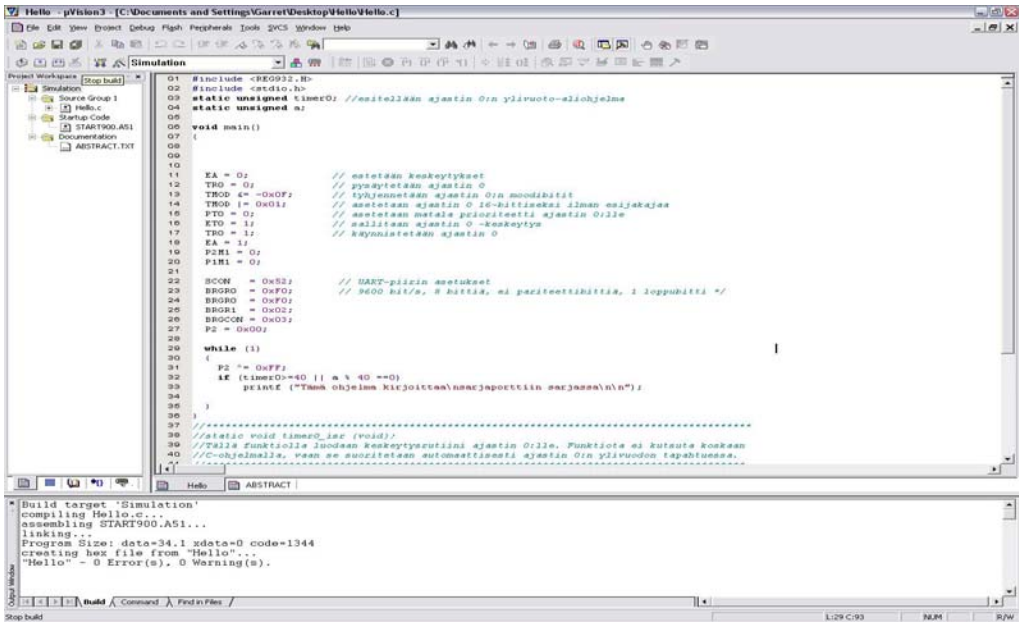


Kuva 12. Keil MCB900 -testialusta [4]

6.1 µVision-ohjelmiston asennus ja yhteyden luominen

µVision-ohjelmiston asennus on melko yksinkertainen ja nopea toimenpide. Asennusikkuna avautuu automaattisesti. kun asennuslevy laitetaan asemaan. Valikosta voidaan valita asennettavat ohjelmat ja lisäksi selata esimerkiksi testialustan tai mikrokontrollerien datalehtiä.

Ohjelmiston asennuksen jälkeen testialusta ja tietokone yhdistetään sarjakaapelilla ja testialustaan kytketään käyttöjännite. Tämän jälkeen µVision-ohjelmasta valitaan käytettävä mikrokontrolleri, minkä jälkeen yhteys on luotu. µVision-ohjelmisto sisältää erillisen monitorointiohjelman, jolla voidaan seurata porttien tiloja ja kaikkia rekisterisisältöjä. Kyseisellä testialustalla monitorointiohjelmaan ei pystynyt kuitenkaan luomaan yhteyttä. Syy saattaa olla ohjelmallinen, sillä muuten korttia pystyi käyttämään ja testaamaan ohjelmiston avulla.



Kuva 13. Näkymä µVision-ohjelman käyttöliittymästä.

Kuvassa 13 on kuva µVision-ohjelman käyttöliittymästä. Kuvan vasemmassa reunassa näkyy kyseiseen projektiin liittyvät tiedostot. Alemman työkalurivin vasemmassa laidassa olevilla napeilla ohjelma voidaan kääntää ja rakentaa. Isossa ruudussa näkyy auki olevan projektin C-kielinen ohjelma, mutta siihen voidaan tuoda esimerkiksi assembly-listaus tai simuloinnissa käytettävä virtuaalinen sarjaportti.

6.2 Testiohjelmat

P89LPC935-mikrokontrollerin toiminnallisia ominaisuuksia testattiin kahdella erilaisella testiohjelmalla.

6.2.1 Ledin vilkutus –ohjelma

Ledin vilkutus –ohjelmalla demonstroidaan mikrokontrollerin porttien ohjausta. Ohjelma vilkuttaa yhtä ledipatsaan lediä. Vilkkumistaajuus on toteutettu ajastinkeskeytyksellä. Ohjelmaa toistetaan, kunnes käyttäjä pysäyttää sen.

```
//ledi.c
#include <REG932.H>      // rekisterien määritys
void main()
{
    EA = 0;           // estetään keskeytykset
    TR0 = 0;          // pysäytetään ajastin 0
    TMOD &= ~0x0F;    // tyhjennetään ajastin 0:n moodibitit
    TMOD |= 0x01;     // asetetaan ajastin 0 16-bittiseksi ilman
                      // esijakajaa
    PT0 = 0;          // asetetaan matala prioriteetti ajastin 0:lle
    ET0 = 1;          // sallitaan ajastin 0 -keskeytys
    TR0 = 1;          // käynnistetään ajastin 0
    EA = 1;           // sallitaa keskeytykset

    P2M1 = 0;
    while(1)                //loppumaton silmukka
    {
    if (timer0<=10)         // mikäli ajastinkeskeytys on tapahtunut
    {                          10 kertaa tai vähemmän
         P2 = 0x01;         // pidetään ledi päällä
    }
    else
    {
      P2 = 0x00;          //muuten sammutetaan ledi
    }
    }
}

//*********************************************************************
//static void timer0_isr (void);
//Tällä funktiolla luodaan keskeytysrutiini ajastin 0:lle. Funktiota ei
kutsuta koskaan
//C-ohjelmalla, vaan se suoritetaan automaattisesti ajastin 0:n
ylivuodon tapahtuessa.
```

```
//*********************************************************************
    static void timer0_isr (void) interrupt 1 using 1
    {

    TR0 = 0;       // pysäytetään ajastin 0
    TR0 = 1;       // käynnistetään ajastin 0 uudelleen
if (timer0>20)    // mikäli keskeytys on tapahtunut yli 20 kertaa,
 {                // nollataan keskeytyslaskuri
    timer0=0;
 }
timer0++;         //kasvatetaan laskurin arvoa yhdellä
}
```

Ohjelman käännös suoritettiin µVision-ohjelmalla. Liitteessä 2 on esitettynä ohjelman käännöslistaus ja liitteessä 4 C-kielestä käännetty Assembly-koodi.

6.2.2 Sarjaporttiin kirjoitus –ohjelma

Ohjelmalla on tarkoitus demonstroida mikrokontrollerin sarjaliikenneominaisuuksia. Sen toimintaa voi tarkastella joko µVision-ohjelmistossa olevan simulointiominaisuuden avulla tai esimerkiksi Microsoft Windowsin mukana tulevalla Hyper Terminal –ohjelmalla. Lisäksi testialustalla olevat ledit vilkkuvat joka kerta, kun sarjaporttiin kirjoitetaan.

```
#include <REG932.H>
#include <stdio.h>
static unsigned a;      // apumuuttuja

void main()
{
  EA = 0;             // estetään keskeytykset
  TR0 = 0;            // pysäytetään ajastin 0
  TMOD &= ~0x0F;      // tyhjennetään ajastin 0:n moodibitit
  TMOD |= 0x01;       // asetetaan ajastin 0 16-bittiseksi ilman
                      //   esijakajaa
  PT0 = 0;            // asetetaan matala prioriteetti ajastin 0:lle
  ET0 = 1;            // sallitaan ajastin 0 -keskeytys
  TR0 = 1;            // käynnistetään ajastin 0
  EA = 1;            // sallitaan keskeytykset
  P2M1 = 0;
  P1M1 = 0;

  SCON   = 0x52;       // alustetaan UART
  BRGR0  = 0xF0;       // 9600 bit/s
  BRGR0  = 0xF0;       // 8 bittinen merkki
  BRGR1  = 0x02;       // ei paritettibittiä
  BRGCON = 0x03;       // 1 loppubitti
  P2 = 0x00;

  while (1)
  {
    P2 ^= 0xFF;
    if (timer0>=40 || a % 40 ==0)
```

```
        printf ("Tämä ohjelma kirjoittaa\nsarjaporttiin sarjassa\n\n");

  }
}
//*********************************************************************
//static void timer0_isr (void);
//Tällä funktiolla luodaan keskeytysrutiini ajastin 0:lle. Funktiota ei
//kutsuta koskaan
//C-ohjelmalla, vaan se suoritetaan automaattisesti ajastin 0:n
yli//vuodon tapahtuessa.
//*********************************************************************
static void timer0_isr (void) interrupt 1 using 1
{

 TR0 = 0;        // pysäytetään ajastin 0
 TR0 = 1;        // käynnistetään ajastin 0 uudelleen
 if (timer0>40 || a>40) // mikäli keskeytys on tapahtunut yli 40
kertaa,
 {                     // nollataan keskeytyslaskuri
timer0=0;
a=0;
 }
timer0++;            //kasvatetaan laskurin arvoa yhdellä
a++;                 //kasvatetaan apumuuttujan arvoa yhdellä
}
```

Ohjelman käännös suoritettiin µVision-ohjelmalla. Liitteessä 3 on esitettynä ohjelman käännöslistaus ja liitteessä 6 C-kielestä käännetty Assembly-koodi.

7 YHTEENVETO

Tutkintotyön tavoitteena oli tutustua mikrokontrollerien rakenteeseen ja toimintatapaan. Erityisesti tarkoituksena oli tutkia Philipsin P89LPC935-mikrokontrollerin rakennetta ja toiminnallisia ominaisuuksia. Tutkimusta tehtiin teoriatasolla tutkimalla valmistajan datalehtiä ja käyttäjän käsikirjaa. Käytännön tutkimustyö toteutettiin tekemällä testiohjelmia, joilla testattiin joitakin mikrokontrollerin ominaisuuksia.

Mikrokontrollerin testaaminen µVision-ohjelmistolla onnistui muuten odotetulla tavalla, mutta erillisen monitorointiohjelman käyttäminen epäonnistui. Monitorointiohjelma ei pystynyt muodostamaan yhteyttä testialustalle, vaikka muuten yhteys tietokoneen ja testialustan välillä toimi. Syy saattoi olla monitorointiohjelman linkkaustiedostossa tai tietokoneen COM-portin asetuksissa. Ongelmanratkaisuun ei kuitenkaan käytetty kohtuuttomasti aikaa, sillä monitorointiohjelman toimivuus ei ollut mikrokontrollerin käytön edellytyksenä.

Monitorointiohjelman puute ei estänyt mikrokontrollerin testaamista, sillä ohjelmat pystyi lataamaan normaalisti mikrokontrollerin muistiin ja ajamaan ne. Tulosten seuranta ja analysointi olisi ollut helpompaa monitorointiohjelman avulla. µVision-ohjelmistossa oleva simulointi-ominaisuus kuitenkin korvasi melko hyvin monitoroinnin

puuttumisen. Simuloinnilla pystyi ajamaan ohjelmia ilman, että niitä tarvitsi ladata mikrokontrollerin muistiin. Tämä helpotti esimerkiksi sarjaliikenneportin tilanseuraamista ja käskynsuoritusjärjestyksen seuraamista.

Tutkintotyön pohjana on käytetty valmistajan, eli Philipsin datalehtiä. Lisäksi valmistajan ilmoittamia ominaisuuksia on testattu tarkoituksiin soveltuvilla ohjelmilla. Tästä tutkintotyöstä on varmasti hyötyä jatkossa, kun halutaan tutustua P89LPC935-mikrokontrolleriin. Työ tarjoaa perustiedot helpossa ja ymmärrettävässä muodossa, joita voi tarvittaessa täydentää valmistajan datalehdistä löytyvillä tiedoilla.

LÄHDELUETTELO

[1] Philips Semiconductors homepage, http://www.semiconductors.philips.com 15.3.2006

[2] PIC Microcontrollers, http://www.mikroelektronika.co.yu/english/product/books/PICbook/1_chapter.htm 15.3.2006

[3] CPU-World, http://www.cpu-world.com 16.3.2006

[4] Keil Embedded Development Tools for ARM7/ARM9/Cortex-M3, XC16x/C16x/ST10, 251, and 8051 Microcontrollers, http://www.keil.com 9.5.2006

[5] Philips Semiconductors – Standard ICs [Home], http://www.standardics.philips.com/ 22.5.2006

LIITTELUETTELO

Liite 1. REG932.H-tiedoston tiedostolistaus 1(3)

```
/*---------------------------------------------
REG932.H

Header file for Philips 89LPC932/935
Copyright (c) 1988-2003 Keil
Elektronik GmbH and Keil Software, Inc.
All rights reserved.
V1.3
---------------------------------------------*/

#ifndef __REG932_H__
#define __REG932_H__

/*  BYTE Registers  */
sfr P0   = 0x80;
sfr P1   = 0x90;
sfr P2   = 0xA0;
sfr P3   = 0xB0;
sfr PSW  = 0xD0;
sfr ACC  = 0xE0;
sfr B    = 0xF0;
sfr SP   = 0x81;
sfr DPL  = 0x82;
sfr DPH  = 0x83;
sfr PCON = 0x87;
sfr TCON = 0x88;
sfr TMOD = 0x89;
sfr TL0  = 0x8A;
sfr TL1  = 0x8B;
sfr TH0  = 0x8C;
sfr TH1  = 0x8D;
sfr IEN0 = 0xA8;
sfr IP0  = 0xB8;
sfr SCON = 0x98;
sfr SBUF = 0x99;
sfr AUXR1  = 0xA2;
sfr SADDR  = 0xA9;
sfr SADEN  = 0xB9;
sfr TL2    = 0xCC;
sfr TH2    = 0xCD;
sfr BRGR0  = 0xBE;
sfr BRGR1  = 0xBF;
sfr BRGCON = 0xBD;
sfr CCCRA  = 0xEA;
sfr CCCRB  = 0xEB;
sfr CCCRC  = 0xEC;
sfr CCCRD  = 0xED;
sfr CMP1   = 0xAC;
sfr CMP2   = 0xAD;
sfr DEECON = 0xF1;
sfr DEEDAT = 0xF2;
sfr DEEADR = 0xF3;
sfr DIVM   = 0x95;
sfr I2ADR  = 0xDB;
sfr I2CON  = 0xD8;
sfr I2DAT  = 0xDA;
sfr I2SCLH = 0xDD;
sfr I2SCLL = 0xDC;
sfr I2STAT = 0xD9;
sfr ICRAH  = 0xAB;
sfr ICRAL  = 0xAA;
sfr ICRBH  = 0xAF;
sfr ICRBL  = 0xAE;
sfr IEN1   = 0xE8;
sfr IP1    = 0xF8;
sfr IP1H   = 0xF7;
sfr KBCON  = 0x94;
sfr KBMASK = 0x86;
sfr KBPATN = 0x93;
sfr OCRAH  = 0xEF;
sfr OCRAL  = 0xEE;
sfr OCRBH  = 0xFB;
sfr OCRBL  = 0xFA;
sfr OCRCH  = 0xFD;
sfr OCRCL  = 0xFC;
sfr OCRDH  = 0xFF;
sfr OCRDL  = 0xFE;
sfr P0M1   = 0x84;
sfr P0M2   = 0x85;
sfr P1M1   = 0x91;
sfr P1M2   = 0x92;
sfr P2M1   = 0xA4;
sfr P2M2   = 0xA5;
```

Liite 1. REG932.H-tiedoston tiedostolistaus 2(3)

```
sfr P3M1   = 0xB1;
sfr P3M2   = 0xB2;
sfr PCONA  = 0xB5;
sfr PT0AD  = 0xF6;
sfr RSTSRC = 0xDF;
sfr RTCCON = 0xD1;
sfr RTCH   = 0xD2;
sfr RTCL   = 0xD3;
sfr SSTAT  = 0xBA;
sfr SPCTL  = 0xE2;
sfr SPSTAT = 0xE1;
sfr SPDAT  = 0xE3;
sfr TAMOD  = 0x8F;
sfr TCR20  = 0xC8;
sfr TCR21  = 0xF9;
sfr TICR2  = 0xC9;
sfr TIFR2  = 0xE9;
sfr TISE2  = 0xDE;
sfr TOR2H  = 0xCF;
sfr TOR2L  = 0xCE;
sfr TPCR2H = 0xCB;
sfr TPCR2L = 0xCA;
sfr TRIM   = 0x96;
sfr WDCON  = 0xA7;
sfr WDL    = 0xC1;
sfr WFEED1 = 0xC2;
sfr WFEED2 = 0xC3;
sfr IP0H   = 0xB7;

/*  BIT Registers  */
/*  PSW   */
sbit CY   = PSW^7;
sbit AC   = PSW^6;
sbit F0   = PSW^5;
sbit RS1  = PSW^4;
sbit RS0  = PSW^3;
sbit OV   = PSW^2;
sbit F1   = PSW^1;
sbit P    = PSW^0;

/*  TCON  */
sbit TF1  = TCON^7;
sbit TR1  = TCON^6;
sbit TF0  = TCON^5;
sbit TR0  = TCON^4;
sbit IE1  = TCON^3;
sbit IT1  = TCON^2;
sbit IE0  = TCON^1;
sbit IT0  = TCON^0;
/*  IEN0   */
sbit EA   = IEN0^7;
sbit EWDRT = IEN0^6;
sbit EBO   = IEN0^5;
sbit ES   = IEN0^4; // alternatively "ESR"
sbit ESR  = IEN0^4;
sbit ET1  = IEN0^3;
sbit EX1  = IEN0^2;
sbit ET0  = IEN0^1;
sbit EX0  = IEN0^0;
/*  IEN1   */
sbit EIEE = IEN1^7;
sbit EST  = IEN1^6;
sbit ECCU = IEN1^4;
sbit ESPI = IEN1^3;
sbit EC   = IEN1^2;
sbit EKBI = IEN1^1;
sbit EI2C = IEN1^0;

/*  IP0   */
sbit PWDRT = IP0^6;
sbit PB0  = IP0^5;
sbit PS   = IP0^4; // alternatively "PSR"
sbit PSR  = IP0^4;
sbit PT1  = IP0^3;
sbit PX1  = IP0^2;
sbit PT0  = IP0^1;
sbit PX0  = IP0^0;

/*  SCON  */
sbit SM0  = SCON^7; // alternatively "FE"
sbit FE   = SCON^7;
sbit SM1  = SCON^6;
sbit SM2  = SCON^5;
sbit REN  = SCON^4;
sbit TB8  = SCON^3;
sbit RB8  = SCON^2;
sbit TI   = SCON^1;
sbit RI   = SCON^0;

/*  I2CON  */
sbit I2EN  = I2CON^6;
sbit STA   = I2CON^5;
sbit STO   = I2CON^4;
```

Liite 1. REG932.H-tiedoston tiedostolistaus 3(3)

```
sbit SI    = I2CON^3;
sbit AA    = I2CON^2;
sbit CRSEL = I2CON^0;

/*  P0  */
sbit KB7 = P0^7; // alternatively "T1"
sbit T1 = P0^7;
sbit KB6 = P0^6; // alternatively "CMP1"
//sbit CMP1 = P0^6;
sbit KB5 = P0^5;
sbit KB4 = P0^4;
sbit KB3 = P0^3;
sbit KB2 = P0^2;
sbit KB1 = P0^1;
sbit KB0 = P0^0; // alternatively "CMP2"
//sbit CMP2 = P0^0;

/*  P1  */
sbit OCC     = P1^7;
sbit OCB     = P1^6;
sbit RST     = P1^5;
sbit INT1    = P1^4;
sbit INT0    = P1^3; // alternatively "SDA"
sbit SDA     = P1^3;
sbit T0      = P1^2; // alternatively "SCL"
sbit SCL     = P1^2;
sbit RxD     = P1^1;
sbit TxD     = P1^0;

/*  P2  */
sbit ICA     = P2^7;
sbit OCA     = P2^6;
sbit SPICLK  = P2^5;
sbit SS      = P2^4;
sbit MISO    = P2^3;
sbit MOSI    = P2^2;
sbit OCD     = P2^1;
sbit ICB     = P2^0;

/*  P3  */
sbit XTAL1= P3^1;
sbit XTAL2= P3^0;

/* TCR20 */
sbit PLLEN = TCR20^7;
sbit HLTRN = TCR20^6;
sbit HLTEN = TCR20^5;
sbit ALTCD = TCR20^4;
```

Liite 2. STDIO.H-tiedoston tiedostolistaus 1(1)

```
/*---------------------------------------------------------------------------
STDIO.H

Prototypes for standard I/O functions.
Copyright (c) 1988-2002 Keil Elektronik GmbH and Keil Software, Inc.
All rights reserved.
---------------------------------------------------------------------------*/

#ifndef __STDIO_H__
#define __STDIO_H__

#ifndef EOF
 #define EOF -1
#endif

#ifndef NULL
 #define NULL ((void *) 0)
#endif

#ifndef _SIZE_T
 #define _SIZE_T
 typedef unsigned int size_t;
#endif

#pragma SAVE
#pragma REGPARMS
extern char _getkey (void);
extern char getchar (void);
extern char ungetchar (char);
extern char putchar (char);
extern int printf   (const char *, ...);
extern int sprintf  (char *, const char *, ...);
extern int vprintf  (const char *, char *);
extern int vsprintf (char *, const char *, char *);
extern char *gets (char *, int n);
extern int scanf (const char *, ...);
extern int sscanf (char *, const char *, ...);
extern int puts (const char *);

#pragma RESTORE

#endif
```

Liite 3. Ledin vilkutus- ja sarjaporttiin kirjoitus –ohjelmien käännöslistaukset 1(1)

Ledin vilkutus –ohjelman käännöslistaus

```
Build target 'Simulator'
compiling ledi.c...
assembling START900.A51...
linking...
Program Size: data=11.0 xdata=0 code=105
creating hex file from "ledi"...
"ledi" - 0 Error(s), 0 Warning(s).
```

Sarjaporttiin kirjoitus –ohjelman käännöslistaus

```
Build target 'Simulation'
compiling sarjaportti.c...
assembling START900.A51...
linking...
Program Size: data=32.1 xdata=0 code=1392
creating hex file from "sarjaportti"...
"sarjaportti" - 0 Error(s), 0 Warning(s).
```

Liite 4. Ledin vilkutus –ohjelman Assembly-koodi 1(2)

```
    251: ?C_STARTUP:     JMP     STARTUP1
   252:
   253:                 RSEG    ?C_C51STARTUP
   254:
   255: STARTUP1:
   256:
   257: IF IDATALEN <> 0
C:0x0000    02005D   LJMP     STARTUP1(C:005D)
C:0x0003    00       NOP
C:0x0004    00       NOP
C:0x0005    00       NOP
C:0x0006    00       NOP
C:0x0007    00       NOP
C:0x0008    00       NOP
C:0x0009    00       NOP
C:0x000A    00       NOP
C:0x000B    0137     AJMP     timer0_isr(C:0037)
     5: void main()
     6: {
     7:
     8:
     9:         EA = 0;            // estetään keskeytykset
C:0x000D    C2AF     CLR      EA(0xA8.7)
    10:     TR0 = 0;             // pysäytetään ajastin 0
C:0x000F    C28C     CLR      TR0(0x88.4)
    11:     TMOD &= ~0x0F;       // tyhjennetään ajastin 0:n moodibitit
C:0x0011    5389F0   ANL      TMOD(0x89),#B(0xF0)
    12:     TMOD |= 0x01;        // asetetaan ajastin 0 16-bittiseksi ilman esijakajaa
C:0x0014    438901   ORL      TMOD(0x89),#0x01
    13:     PT0 = 0;             // asetetaan matala prioriteetti ajastin 0:lle
C:0x0017    C2B9     CLR      PT0(0xB8.1)
    14:         ET0 = 1;           // sallitaan ajastin 0 -keskeytys
C:0x0019    D2A9     SETB     ET0(0xA8.1)
    15:     TR0 = 1;             // käynnistetään ajastin 0
C:0x001B    D28C     SETB     TR0(0x88.4)
    16:     EA = 1;              // sallitaa keskeytykset
    17:
C:0x001D    D2AF     SETB     EA(0xA8.7)
    18:         P2M1 = 0;
C:0x001F    E4       CLR      A
C:0x0020    F5A4     MOV      P2M1(0xA4),A
    19:         while(1)              //loppumaton silmukka
    20:     {
    21:     if (timer0<=10)          // mikäli ajastinkeskeytys on tapahtunut 10 kertaa tai
vähemmän
C:0x0022    D3       SETB     C
C:0x0023    E509     MOV      A,0x09
C:0x0025    940A     SUBB     A,#0x0A
C:0x0027    E508     MOV      A,0x08
C:0x0029    9400     SUBB     A,#0x00
C:0x002B    5005     JNC      C:0032
    22:     {
    23:           P2 = 0x01;    // pidetään ledi päällä
C:0x002D    75A001   MOV      P2(0xA0),#0x01
    24:     }
    25:     else
C:0x0030    80F0     SJMP     C:0022
    26:     {
    27:       P2 = 0x00;            //muuten sammutetaan ledi

C:0x0032    E4       CLR      A
C:0x0033    F5A0     MOV      P2(0xA0),A
    28:     }
    29:     }
    30: }
    31:
```

Liite 4. Ledin vilkutus –ohjelman Assembly-koodi 2(2)

```
//*****************************************************************************
    32:     //static void timer0_isr (void);
    33:     //Tällä funktiolla luodaan keskeytysrutiini ajastin 0:lle. Funktiota ei kutsuta
koskaan
    34:         //C-ohjelmalla, vaan se suoritetaan automaattisesti ajastin 0:n ylivuodon
tapahtuessa.
    35:
//*****************************************************************************
C:0x0035    80EB     SJMP     C:0022
    36:     static void timer0_isr (void) interrupt 1 using 1
    37:     {
    38:
C:0x0037    C0E0     PUSH     ACC(0xE0)
C:0x0039    C0D0     PUSH     PSW(0xD0)
    39:     TR0 = 0;        // pysäytetään ajastin 0
C:0x003B    C28C     CLR      TR0(0x88.4)
    40:     TR0 = 1;        // käynnistetään ajastin 0 uudelleen
C:0x003D    D28C     SETB     TR0(0x88.4)
    41:         if (timer0>20)  // mikäli keskeytys on tapahtunut yli 20 kertaa,
C:0x003F    D3       SETB     C
C:0x0040    E509     MOV      A,0x09
C:0x0042    9414     SUBB     A,#0x14
C:0x0044    E508     MOV      A,0x08
C:0x0046    9400     SUBB     A,#0x00
C:0x0048    4006     JC       C:0050
    42:         {                           // nollataan keskeytyslaskuri
    43:         timer0=0;
C:0x004A    750800   MOV      0x08,#0x00
C:0x004D    750900   MOV      0x09,#0x00
    44:         }
    45:         timer0++;               //kasvatetaan laskurin arvoa yhdellä
C:0x0050    0509     INC      0x09
C:0x0052    E509     MOV      A,0x09
C:0x0054    7002     JNZ      C:0058
C:0x0056    0508     INC      0x08
    46:         }
C:0x0058    D0D0     POP      PSW(0xD0)
C:0x005A    D0E0     POP      ACC(0xE0)
C:0x005C    32       RETI
   258:              MOV     R0,#IDATALEN - 1
C:0x005D    78FF     MOV      R0,#OCRDH(0xFF)
   259:              CLR     A
C:0x005F    E4       CLR      A
   260: IDATALOOP:   MOV     @R0,A
C:0x0060    F6       MOV      @R0,A
   261:              DJNZ    R0,IDATALOOP
C:0x0061    D8FD     DJNZ     R0,IDATALOOP(C:0060)
   299:              MOV     SP,#?STACK-1
C:0x0063    758109   MOV      SP(0x81),#0x09
   300:              JMP     ?C_START
C:0x0066    02000D   LJMP     main(C:000D)
```

Liite 5. Sarjaporttiin kirjoitus –ohjelman Assembly-koodi 1(12)

```
   251: ?C_STARTUP:     JMP     STARTUP1
   252:
   253:                 RSEG    ?C_C51STARTUP
   254:
   255: STARTUP1:
   256:
   257: IF IDATALEN <> 0
C:0x0000    020530   LJMP     STARTUP1(C:0530)
C:0x0003    00       NOP
C:0x0004    00       NOP
C:0x0005    00       NOP
C:0x0006    00       NOP
C:0x0007    00       NOP
C:0x0008    00       NOP
C:0x0009    00       NOP
C:0x000A    00       NOP
C:0x000B    819A     AJMP     timer0_isr(C:049A)
C:0x000D    E517     MOV      A,0x17
C:0x000F    240B     ADD      A,#0x0B
C:0x0011    F8       MOV      R0,A
C:0x0012    E6       MOV      A,@R0
C:0x0013    0517     INC      0x17
C:0x0015    22       RET
C:0x0016    7808     MOV      R0,#?_PRINTF517?BYTE(0x08)
C:0x0018    300702   JNB      0x20.7,C:001D
C:0x001B    780B     MOV      R0,#0x0B
C:0x001D    E4       CLR      A
C:0x001E    75F001   MOV      B(0xF0),#0x01
C:0x0021    120416   LCALL    C?PLDIIDATA(C:0416)
C:0x0024    0203BE   LJMP     C?CLDPTR(C:03BE)
C:0x0027    2000EB   JB       0x20.0,C:0015
C:0x002A    7F2E     MOV      R7,#0x2E
C:0x002C    D200     SETB     0x20.0
C:0x002E    8018     SJMP     C:0048
C:0x0030    EF       MOV      A,R7
C:0x0031    540F     ANL      A,#0x0F
C:0x0033    2490     ADD      A,#P1(0x90)
C:0x0035    D4       DA       A
C:0x0036    3440     ADDC     A,#0x40
C:0x0038    D4       DA       A
C:0x0039    FF       MOV      R7,A
C:0x003A    30040B   JNB      0x20.4,C:0048
C:0x003D    EF       MOV      A,R7
C:0x003E    24BF     ADD      A,#BRGR1(0xBF)
C:0x0040    B41A00   CJNE     A,#0x1A,C:0043
C:0x0043    5003     JNC      C:0048
C:0x0045    2461     ADD      A,#0x61
C:0x0047    FF       MOV      R7,A
C:0x0048    E518     MOV      A,0x18
C:0x004A    6002     JZ       C:004E
C:0x004C    1518     DEC      0x18
C:0x004E    051B     INC      0x1B
C:0x0050    E51B     MOV      A,0x1B
C:0x0052    7002     JNZ      C:0056
C:0x0054    051A     INC      0x1A
C:0x0056    30070D   JNB      0x20.7,C:0066
C:0x0059    7808     MOV      R0,#?_PRINTF517?BYTE(0x08)
C:0x005B    E4       CLR      A
C:0x005C    75F001   MOV      B(0xF0),#0x01
C:0x005F    120416   LCALL    C?PLDIIDATA(C:0416)
C:0x0062    EF       MOV      A,R7
C:0x0063    020404   LJMP     C?CSTPTR(C:0404)
C:0x0066    020509   LJMP     PUTCHAR(C:0509)
                SPRINTF:
C:0x0069    7403     MOV      A,#0x03
C:0x006B    D207     SETB     0x20.7
C:0x006D    8003     SJMP     C:0072
                PRINTF:
C:0x006F    E4       CLR      A
C:0x0070    C207     CLR      0x20.7
C:0x0072    F517     MOV      0x17,A
```

```
C:0x0074    8B08     MOV      ?_PRINTF517?BYTE(0x08),R3
C:0x0076    8A09     MOV      0x09,R2
C:0x0078    890A     MOV      0x0A,R1
C:0x007A    E4       CLR      A
C:0x007B    F518     MOV      0x18,A
C:0x007D    F51A     MOV      0x1A,A
C:0x007F    F51B     MOV      0x1B,A
C:0x0081    E518     MOV      A,0x18
C:0x0083    6007     JZ       C:008C
C:0x0085    7F20     MOV      R7,#0x20
C:0x0087    120048   LCALL    C:0048
C:0x008A    80F5     SJMP     C:0081
C:0x008C    7519FF   MOV      0x19,#OCRDH(0xFF)
C:0x008F    C201     CLR      0x20.1
C:0x0091    C200     CLR      0x20.0
C:0x0093    C202     CLR      0x20.2
C:0x0095    C203     CLR      0x20.3
C:0x0097    C205     CLR      0x20.5
C:0x0099    C206     CLR      0x20.6
C:0x009B    C208     CLR      0x21.0
C:0x009D    120016   LCALL    C:0016
C:0x00A0    FF       MOV      R7,A
C:0x00A1    700D     JNZ      C:00B0
C:0x00A3    300705   JNB      0x20.7,C:00AB
C:0x00A6    7F00     MOV      R7,#0x00
C:0x00A8    120059   LCALL    C:0059
C:0x00AB    AF1B     MOV      R7,0x1B
C:0x00AD    AE1A     MOV      R6,0x1A
C:0x00AF    22       RET
C:0x00B0    B4255F   CJNE     A,#0x25,C:0112
C:0x00B3    C2D5     CLR      F0(0xD0.5)
C:0x00B5    C204     CLR      0x20.4
C:0x00B7    120016   LCALL    C:0016
C:0x00BA    FF       MOV      R7,A
C:0x00BB    24D0     ADD      A,#PSW(0xD0)
C:0x00BD    B40A00   CJNE     A,#0x0A,C:00C0
C:0x00C0    501A     JNC      C:00DC
C:0x00C2    75F00A   MOV      B(0xF0),#0x0A
C:0x00C5    7818     MOV      R0,#0x18
C:0x00C7    30D505   JNB      F0(0xD0.5),C:00CF
C:0x00CA    08       INC      R0
C:0x00CB    B6FF01   CJNE     @R0,#OCRDH(0xFF),C:00CF
C:0x00CE    06       INC      @R0
C:0x00CF    C6       XCH      A,@R0
C:0x00D0    A4       MUL      AB
C:0x00D1    26       ADD      A,@R0
C:0x00D2    F6       MOV      @R0,A
C:0x00D3    20D504   JB       F0(0xD0.5),C:00DA
C:0x00D6    7002     JNZ      C:00DA
C:0x00D8    D203     SETB     0x20.3
C:0x00DA    80D9     SJMP     C:00B5
C:0x00DC    24CF     ADD      A,#TOR2H(0xCF)
C:0x00DE    B41A00   CJNE     A,#0x1A,C:00E1
C:0x00E1    EF       MOV      A,R7
C:0x00E2    5004     JNC      C:00E8
C:0x00E4    C2E5     CLR      0xE0.5
C:0x00E6    D204     SETB     0x20.4
C:0x00E8    020259   LJMP     C:0259
C:0x00EB    D201     SETB     0x20.1
C:0x00ED    80C6     SJMP     C:00B5
C:0x00EF    D200     SETB     0x20.0
C:0x00F1    80C0     SJMP     C:00B3
C:0x00F3    D202     SETB     0x20.2
C:0x00F5    80BC     SJMP     C:00B3
C:0x00F7    D2D5     SETB     F0(0xD0.5)
C:0x00F9    80BA     SJMP     C:00B5
C:0x00FB    D205     SETB     0x20.5
C:0x00FD    80B4     SJMP     C:00B3
C:0x00FF    7F20     MOV      R7,#0x20
C:0x0101    120048   LCALL    C:0048
C:0x0104    200207   JB       0x20.2,C:010E
```

```
C:0x0107    7401     MOV      A,#0x01
C:0x0109    B51800   CJNE     A,0x18,C:010C
C:0x010C    40F1     JC       C:00FF
C:0x010E    12000D   LCALL    C:000D
C:0x0111    FF       MOV      R7,A
C:0x0112    120048   LCALL    C:0048
C:0x0115    020081   LJMP     C:0081
C:0x0118    D208     SETB     0x21.0
C:0x011A    D206     SETB     0x20.6
C:0x011C    8095     SJMP     C:00B3
C:0x011E    12000D   LCALL    C:000D
C:0x0121    FB       MOV      R3,A
C:0x0122    12000D   LCALL    C:000D
C:0x0125    FA       MOV      R2,A
C:0x0126    12000D   LCALL    C:000D
C:0x0129    F9       MOV      R1,A
C:0x012A    4A       ORL      A,R2
C:0x012B    4B       ORL      A,R3
C:0x012C    7006     JNZ      C:0134
C:0x012E    792A     MOV      R1,#0x2A
C:0x0130    7A03     MOV      R2,#0x03
C:0x0132    7BFF     MOV      R3,#OCRDH(0xFF)
C:0x0134    20022E   JB       0x20.2,C:0165
C:0x0137    E518     MOV      A,0x18
C:0x0139    602A     JZ       C:0165
C:0x013B    7E00     MOV      R6,#0x00
C:0x013D    8E82     MOV      DPL(0x82),R6
C:0x013F    758300   MOV      DPH(0x83),#0x00
C:0x0142    1203D7   LCALL    C?CLDOPTR(C:03D7)
C:0x0145    6006     JZ       C:014D
C:0x0147    0E       INC      R6
C:0x0148    EE       MOV      A,R6
C:0x0149    6519     XRL      A,0x19
C:0x014B    70F0     JNZ      C:013D
C:0x014D    C2D5     CLR      F0(0xD0.5)
C:0x014F    EB       MOV      A,R3
C:0x0150    C0E0     PUSH     ACC(0xE0)
C:0x0152    EA       MOV      A,R2
C:0x0153    C0E0     PUSH     ACC(0xE0)
C:0x0155    E9       MOV      A,R1
C:0x0156    C0E0     PUSH     ACC(0xE0)
C:0x0158    EE       MOV      A,R6
C:0x0159    1202A0   LCALL    C:02A0
C:0x015C    D0E0     POP      ACC(0xE0)
C:0x015E    F9       MOV      R1,A
C:0x015F    D0E0     POP      ACC(0xE0)
C:0x0161    FA       MOV      R2,A
C:0x0162    D0E0     POP      ACC(0xE0)
C:0x0164    FB       MOV      R3,A
C:0x0165    1203BE   LCALL    C?CLDPTR(C:03BE)
C:0x0168    FF       MOV      R7,A
C:0x0169    60AA     JZ       C:0115
C:0x016B    EB       MOV      A,R3
C:0x016C    C0E0     PUSH     ACC(0xE0)
C:0x016E    EA       MOV      A,R2
C:0x016F    C0E0     PUSH     ACC(0xE0)
C:0x0171    E9       MOV      A,R1
C:0x0172    C0E0     PUSH     ACC(0xE0)
C:0x0174    120048   LCALL    C:0048
C:0x0177    D0E0     POP      ACC(0xE0)
C:0x0179    2401     ADD      A,#0x01
C:0x017B    F9       MOV      R1,A
C:0x017C    D0E0     POP      ACC(0xE0)
C:0x017E    3400     ADDC     A,#0x00
C:0x0180    FA       MOV      R2,A
C:0x0181    D0E0     POP      ACC(0xE0)
C:0x0183    FB       MOV      R3,A
C:0x0184    E519     MOV      A,0x19
C:0x0186    04       INC      A
C:0x0187    60DC     JZ       C:0165
C:0x0189    D519D9   DJNZ     0x19,C:0165
```

```
C:0x018C    8087     SJMP     C:0115
C:0x018E    7BFF     MOV      R3,#OCRDH(0xFF)
C:0x0190    7A02     MOV      R2,#0x02
C:0x0192    799C     MOV      R1,#0x9C
C:0x0194    D202     SETB     0x20.2
C:0x0196    809C     SJMP     C:0134
C:0x0198    7910     MOV      R1,#0x10
C:0x019A    8002     SJMP     C:019E
C:0x019C    7908     MOV      R1,#?_PRINTF517?BYTE(0x08)
C:0x019E    C206     CLR      0x20.6
C:0x01A0    C208     CLR      0x21.0
C:0x01A2    8008     SJMP     C:01AC
C:0x01A4    D2D5     SETB     F0(0xD0.5)
C:0x01A6    790A     MOV      R1,#0x0A
C:0x01A8    8004     SJMP     C:01AE
C:0x01AA    790A     MOV      R1,#0x0A
C:0x01AC    C2D5     CLR      F0(0xD0.5)
C:0x01AE    E519     MOV      A,0x19
C:0x01B0    04       INC      A
C:0x01B1    7002     JNZ      C:01B5
C:0x01B3    F519     MOV      0x19,A
C:0x01B5    E4       CLR      A
C:0x01B6    FA       MOV      R2,A
C:0x01B7    FD       MOV      R5,A
C:0x01B8    FE       MOV      R6,A
C:0x01B9    FF       MOV      R7,A
C:0x01BA    12000D   LCALL    C:000D
C:0x01BD    FC       MOV      R4,A
C:0x01BE    7B08     MOV      R3,#?_PRINTF517?BYTE(0x08)
C:0x01C0    200113   JB       0x20.1,C:01D6
C:0x01C3    12000D   LCALL    C:000D
C:0x01C6    FD       MOV      R5,A
C:0x01C7    7B10     MOV      R3,#0x10
C:0x01C9    30000A   JNB      0x20.0,C:01D6
C:0x01CC    12000D   LCALL    C:000D
C:0x01CF    FE       MOV      R6,A
C:0x01D0    12000D   LCALL    C:000D
C:0x01D3    FF       MOV      R7,A
C:0x01D4    7B20     MOV      R3,#0x20
C:0x01D6    EC       MOV      A,R4
C:0x01D7    33       RLC      A
C:0x01D8    82D5     ANL      C,F0(0xD0.5)
C:0x01DA    92D5     MOV      F0(0xD0.5),C
C:0x01DC    5013     JNC      C:01F1
C:0x01DE    C3       CLR      C
C:0x01DF    E4       CLR      A
C:0x01E0    300006   JNB      0x20.0,C:01E9
C:0x01E3    9F       SUBB     A,R7
C:0x01E4    FF       MOV      R7,A
C:0x01E5    E4       CLR      A
C:0x01E6    9E       SUBB     A,R6
C:0x01E7    FE       MOV      R6,A
C:0x01E8    E4       CLR      A
C:0x01E9    200103   JB       0x20.1,C:01EF
C:0x01EC    9D       SUBB     A,R5
C:0x01ED    FD       MOV      R5,A
C:0x01EE    E4       CLR      A
C:0x01EF    9C       SUBB     A,R4
C:0x01F0    FC       MOV      R4,A
C:0x01F1    E4       CLR      A
C:0x01F2    CB       XCH      A,R3
C:0x01F3    F8       MOV      R0,A
C:0x01F4    C201     CLR      0x20.1
C:0x01F6    EC       MOV      A,R4
C:0x01F7    700C     JNZ      C:0205
C:0x01F9    CF       XCH      A,R7
C:0x01FA    CE       XCH      A,R6
C:0x01FB    CD       XCH      A,R5
C:0x01FC    CC       XCH      A,R4
C:0x01FD    E8       MOV      A,R0
C:0x01FE    24F8     ADD      A,#IP1(0xF8)
```

```
C:0x0200    F8       MOV      R0,A
C:0x0201    70F3     JNZ      C:01F6
C:0x0203    8017     SJMP     C:021C
C:0x0205    C3       CLR      C
C:0x0206    EF       MOV      A,R7
C:0x0207    33       RLC      A
C:0x0208    FF       MOV      R7,A
C:0x0209    EE       MOV      A,R6
C:0x020A    33       RLC      A
C:0x020B    FE       MOV      R6,A
C:0x020C    ED       MOV      A,R5
C:0x020D    33       RLC      A
C:0x020E    FD       MOV      R5,A
C:0x020F    EC       MOV      A,R4
C:0x0210    33       RLC      A
C:0x0211    FC       MOV      R4,A
C:0x0212    EB       MOV      A,R3
C:0x0213    33       RLC      A
C:0x0214    FB       MOV      R3,A
C:0x0215    99       SUBB     A,R1
C:0x0216    4002     JC       C:021A
C:0x0218    FB       MOV      R3,A
C:0x0219    0F       INC      R7
C:0x021A    D8E9     DJNZ     R0,C:0205
C:0x021C    EB       MOV      A,R3
C:0x021D    300105   JNB      0x20.1,C:0225
C:0x0220    F8       MOV      R0,A
C:0x0221    D0E0     POP      ACC(0xE0)
C:0x0223    C4       SWAP     A
C:0x0224    48       ORL      A,R0
C:0x0225    B201     CPL      0x20.1
C:0x0227    C0E0     PUSH     ACC(0xE0)
C:0x0229    0A       INC      R2
C:0x022A    EC       MOV      A,R4
C:0x022B    4D       ORL      A,R5
C:0x022C    4E       ORL      A,R6
C:0x022D    4F       ORL      A,R7
C:0x022E    7820     MOV      R0,#0x20
C:0x0230    7B00     MOV      R3,#0x00
C:0x0232    70C2     JNZ      C:01F6
C:0x0234    EA       MOV      A,R2
C:0x0235    B51900   CJNE     A,0x19,C:0238
C:0x0238    40BC     JC       C:01F6
C:0x023A    C0E0     PUSH     ACC(0xE0)
C:0x023C    1202A2   LCALL    C:02A2
C:0x023F    D0F0     POP      B(0xF0)
C:0x0241    D0E0     POP      ACC(0xE0)
C:0x0243    200104   JB       0x20.1,C:024A
C:0x0246    C4       SWAP     A
C:0x0247    C0E0     PUSH     ACC(0xE0)
C:0x0249    C4       SWAP     A
C:0x024A    B201     CPL      0x20.1
C:0x024C    C0F0     PUSH     B(0xF0)
C:0x024E    120031   LCALL    C:0031
C:0x0251    D0F0     POP      B(0xF0)
C:0x0253    D5F0EB   DJNZ     B(0xF0),C:0241
C:0x0256    020081   LJMP     C:0081
C:0x0259    120426   LCALL    C?CCASE(C:0426)
C:0x025C    011E     AJMP     C:001E
C:0x025E    530198   ANL      0x01,#SCON(0x98)
C:0x0261    58       ANL      A,R0
C:0x0262    00       NOP
C:0x0263    EF       MOV      A,R7
C:0x0264    4C       ORL      A,R4
C:0x0265    00       NOP
C:0x0266    EB       MOV      A,R3
C:0x0267    4201     ORL      0x01,A
C:0x0269    9C       SUBB     A,R4
C:0x026A    4F       ORL      A,R7
C:0x026B    01A4     AJMP     C:00A4
C:0x026D    4401     ORL      A,#0x01
```

```
C:0x026F    A4       MUL      AB
C:0x0270    49       ORL      A,R1
C:0x0271    0104     AJMP     C:0004
C:0x0273    4301AA   ORL      0x01,#ICRAL(0xAA)
C:0x0276    5501     ANL      A,0x01
C:0x0278    8E46     MOV      0x46,R6
C:0x027A    018E     AJMP     C:008E
C:0x027C    4501     ORL      A,0x01
C:0x027E    8E47     MOV      0x47,R6
C:0x0280    03       RR       A
C:0x0281    4A       ORL      A,R2
C:0x0282    5000     JNC      C:0284
C:0x0284    F3       MOVX     @R1,A
C:0x0285    2D       ADD      A,R5
C:0x0286    00       NOP
C:0x0287    F7       MOV      @R1,A
C:0x0288    2E       ADD      A,R6
C:0x0289    011A     AJMP     C:001A
C:0x028B    2B       ADD      A,R3
C:0x028C    00       NOP
C:0x028D    FB       MOV      R3,A
C:0x028E    23       RL       A
C:0x028F    0118     AJMP     C:0018
C:0x0291    200333   JB       0x20.3,C:02C7
C:0x0294    2A       ADD      A,R2
C:0x0295    00       NOP
C:0x0296    B3       CPL      C
C:0x0297    48       ORL      A,R0
C:0x0298    00       NOP
C:0x0299    00       NOP
C:0x029A    0112     AJMP     C:0012
C:0x029C    3F       ADDC     A,R7
C:0x029D    3F       ADDC     A,R7
C:0x029E    3F       ADDC     A,R7
C:0x029F    00       NOP
C:0x02A0    790A     MOV      R1,#0x0A
C:0x02A2    A2D5     MOV      C,F0(0xD0.5)
C:0x02A4    200314   JB       0x20.3,C:02BB
C:0x02A7    300509   JNB      0x20.5,C:02B3
C:0x02AA    B91002   CJNE     R1,#0x10,C:02AF
C:0x02AD    04       INC      A
C:0x02AE    04       INC      A
C:0x02AF    B90801   CJNE     R1,#?_PRINTF517?BYTE(0x08),C:02B3
C:0x02B2    04       INC      A
C:0x02B3    A2D5     MOV      C,F0(0xD0.5)
C:0x02B5    200602   JB       0x20.6,C:02BA
C:0x02B8    5001     JNC      C:02BB
C:0x02BA    04       INC      A
C:0x02BB    200268   JB       0x20.2,C:0326
C:0x02BE    9202     MOV      0x20.2,C
C:0x02C0    B51800   CJNE     A,0x18,C:02C3
C:0x02C3    5034     JNC      C:02F9
C:0x02C5    C0E0     PUSH     ACC(0xE0)
C:0x02C7    7F20     MOV      R7,#0x20
C:0x02C9    300319   JNB      0x20.3,C:02E5
C:0x02CC    7F30     MOV      R7,#0x30
C:0x02CE    A202     MOV      C,0x20.2
C:0x02D0    7206     ORL      C,0x20.6
C:0x02D2    7205     ORL      C,0x20.5
C:0x02D4    500F     JNC      C:02E5
C:0x02D6    1202F9   LCALL    C:02F9
C:0x02D9    C202     CLR      0x20.2
C:0x02DB    C206     CLR      0x20.6
C:0x02DD    C205     CLR      0x20.5
C:0x02DF    C208     CLR      0x21.0
C:0x02E1    7F30     MOV      R7,#0x30
C:0x02E3    800F     SJMP     C:02F4
C:0x02E5    300503   JNB      0x20.5,C:02EB
C:0x02E8    E9       MOV      A,R1
C:0x02E9    C0E0     PUSH     ACC(0xE0)
C:0x02EB    120048   LCALL    C:0048
```

```
C:0x02EE    300503   JNB      0x20.5,C:02F4
C:0x02F1    D0E0     POP      ACC(0xE0)
C:0x02F3    F9       MOV      R1,A
C:0x02F4    D0E0     POP      ACC(0xE0)
C:0x02F6    B518CC   CJNE     A,0x18,C:02C5
C:0x02F9    300517   JNB      0x20.5,C:0313
C:0x02FC    7F30     MOV      R7,#0x30
C:0x02FE    B9100C   CJNE     R1,#0x10,C:030D
C:0x0301    120048   LCALL    C:0048
C:0x0304    7F58     MOV      R7,#0x58
C:0x0306    300407   JNB      0x20.4,C:0310
C:0x0309    7F78     MOV      R7,#0x78
C:0x030B    8003     SJMP     C:0310
C:0x030D    B90803   CJNE     R1,#?_PRINTF517?BYTE(0x08),C:0313
C:0x0310    120048   LCALL    C:0048
C:0x0313    300205   JNB      0x20.2,C:031B
C:0x0316    7F2D     MOV      R7,#0x2D
C:0x0318    020048   LJMP     C:0048
C:0x031B    7F20     MOV      R7,#0x20
C:0x031D    2008F8   JB       0x21.0,C:0318
C:0x0320    7F2B     MOV      R7,#0x2B
C:0x0322    2006F3   JB       0x20.6,C:0318
C:0x0325    22       RET
C:0x0326    9202     MOV      0x20.2,C
C:0x0328    80CF     SJMP     C:02F9
C:0x032A    28       ADD      A,R0
C:0x032B    6E       XRL      A,R6
C:0x032C    756C6C   MOV      0x6C,#0x6C
C:0x032F    29       ADD      A,R1
C:0x0330    00       NOP
C:0x0331    D201     SETB     0x20.1
C:0x0333    12000D   LCALL    C:000D
C:0x0336    3001F8   JNB      0x20.1,C:0331
C:0x0339    C201     CLR      0x20.1
C:0x033B    7818     MOV      R0,#0x18
C:0x033D    30D501   JNB      F0(0xD0.5),C:0341
C:0x0340    08       INC      R0
C:0x0341    F6       MOV      @R0,A
C:0x0342    0200B3   LJMP     C:00B3
C:0x0345    2D       ADD      A,R5
C:0x0346    5043     JNC      C:038B
C:0x0348    49       ORL      A,R1
C:0x0349    58       ANL      A,R0
C:0x034A    12000D   LCALL    C:000D
C:0x034D    2403     ADD      A,#0x03
C:0x034F    B40500   CJNE     A,#0x05,C:0352
C:0x0352    4001     JC       C:0355
C:0x0354    E4       CLR      A
C:0x0355    900345   MOV      DPTR,#0x0345
C:0x0358    93       MOVC     A,@A+DPTR
C:0x0359    120039   LCALL    C:0039
C:0x035C    743A     MOV      A,#0x3A
C:0x035E    120039   LCALL    C:0039
C:0x0361    D203     SETB     0x20.3
C:0x0363    751804   MOV      0x18,#0x04
C:0x0366    020198   LJMP     C:0198
                C?UIDIV:
C:0x0369    BC000B   CJNE     R4,#0x00,C:0377
C:0x036C    BE0029   CJNE     R6,#0x00,C:0398
C:0x036F    EF       MOV      A,R7
C:0x0370    8DF0     MOV      B(0xF0),R5
C:0x0372    84       DIV      AB
C:0x0373    FF       MOV      R7,A
C:0x0374    ADF0     MOV      R5,B(0xF0)
C:0x0376    22       RET
C:0x0377    E4       CLR      A
C:0x0378    CC       XCH      A,R4
C:0x0379    F8       MOV      R0,A
C:0x037A    75F008   MOV      B(0xF0),#?_PRINTF517?BYTE(0x08)
C:0x037D    EF       MOV      A,R7
C:0x037E    2F       ADD      A,R7
```

```
C:0x037F    FF       MOV      R7,A
C:0x0380    EE       MOV      A,R6
C:0x0381    33       RLC      A
C:0x0382    FE       MOV      R6,A
C:0x0383    EC       MOV      A,R4
C:0x0384    33       RLC      A
C:0x0385    FC       MOV      R4,A
C:0x0386    EE       MOV      A,R6
C:0x0387    9D       SUBB     A,R5
C:0x0388    EC       MOV      A,R4
C:0x0389    98       SUBB     A,R0
C:0x038A    4005     JC       C:0391
C:0x038C    FC       MOV      R4,A
C:0x038D    EE       MOV      A,R6
C:0x038E    9D       SUBB     A,R5
C:0x038F    FE       MOV      R6,A
C:0x0390    0F       INC      R7
C:0x0391    D5F0E9   DJNZ     B(0xF0),C:037D
C:0x0394    E4       CLR      A
C:0x0395    CE       XCH      A,R6
C:0x0396    FD       MOV      R5,A
C:0x0397    22       RET
C:0x0398    ED       MOV      A,R5
C:0x0399    F8       MOV      R0,A
C:0x039A    F5F0     MOV      B(0xF0),A
C:0x039C    EE       MOV      A,R6
C:0x039D    84       DIV      AB
C:0x039E    20D21C   JB       OV(0xD0.2),C:03BD
C:0x03A1    FE       MOV      R6,A
C:0x03A2    ADF0     MOV      R5,B(0xF0)
C:0x03A4    75F008   MOV      B(0xF0),#?_PRINTF517?BYTE(0x08)
C:0x03A7    EF       MOV      A,R7
C:0x03A8    2F       ADD      A,R7
C:0x03A9    FF       MOV      R7,A
C:0x03AA    ED       MOV      A,R5
C:0x03AB    33       RLC      A
C:0x03AC    FD       MOV      R5,A
C:0x03AD    4007     JC       C:03B6
C:0x03AF    98       SUBB     A,R0
C:0x03B0    5006     JNC      C:03B8
C:0x03B2    D5F0F2   DJNZ     B(0xF0),C:03A7
C:0x03B5    22       RET
C:0x03B6    C3       CLR      C
C:0x03B7    98       SUBB     A,R0
C:0x03B8    FD       MOV      R5,A
C:0x03B9    0F       INC      R7
C:0x03BA    D5F0EA   DJNZ     B(0xF0),C:03A7
C:0x03BD    22       RET
                C?CLDPTR:
C:0x03BE    BB0106   CJNE     R3,#0x01,C:03C7
C:0x03C1    8982     MOV      DPL(0x82),R1
C:0x03C3    8A83     MOV      DPH(0x83),R2
C:0x03C5    E0       MOVX     A,@DPTR
C:0x03C6    22       RET
C:0x03C7    5002     JNC      C:03CB
C:0x03C9    E7       MOV      A,@R1
C:0x03CA    22       RET
C:0x03CB    BBFE02   CJNE     R3,#OCRDL(0xFE),C:03D0
C:0x03CE    E3       MOVX     A,@R1
C:0x03CF    22       RET
C:0x03D0    8982     MOV      DPL(0x82),R1
C:0x03D2    8A83     MOV      DPH(0x83),R2
C:0x03D4    E4       CLR      A
C:0x03D5    93       MOVC     A,@A+DPTR
C:0x03D6    22       RET
                C?CLDOPTR:
C:0x03D7    BB010C   CJNE     R3,#0x01,C:03E6
C:0x03DA    E582     MOV      A,DPL(0x82)
C:0x03DC    29       ADD      A,R1
C:0x03DD    F582     MOV      DPL(0x82),A
C:0x03DF    E583     MOV      A,DPH(0x83)
```

```
C:0x03E1    3A       ADDC     A,R2
C:0x03E2    F583     MOV      DPH(0x83),A
C:0x03E4    E0       MOVX     A,@DPTR
C:0x03E5    22       RET
C:0x03E6    5006     JNC      C:03EE
C:0x03E8    E9       MOV      A,R1
C:0x03E9    2582     ADD      A,DPL(0x82)
C:0x03EB    F8       MOV      R0,A
C:0x03EC    E6       MOV      A,@R0
C:0x03ED    22       RET
C:0x03EE    BBFE06   CJNE     R3,#OCRDL(0xFE),C:03F7
C:0x03F1    E9       MOV      A,R1
C:0x03F2    2582     ADD      A,DPL(0x82)
C:0x03F4    F8       MOV      R0,A
C:0x03F5    E2       MOVX     A,@R0
C:0x03F6    22       RET
C:0x03F7    E582     MOV      A,DPL(0x82)
C:0x03F9    29       ADD      A,R1
C:0x03FA    F582     MOV      DPL(0x82),A
C:0x03FC    E583     MOV      A,DPH(0x83)
C:0x03FE    3A       ADDC     A,R2
C:0x03FF    F583     MOV      DPH(0x83),A
C:0x0401    E4       CLR      A
C:0x0402    93       MOVC     A,@A+DPTR
C:0x0403    22       RET
                C?CSTPTR:
C:0x0404    BB0106   CJNE     R3,#0x01,C:040D
C:0x0407    8982     MOV      DPL(0x82),R1
C:0x0409    8A83     MOV      DPH(0x83),R2
C:0x040B    F0       MOVX     @DPTR,A
C:0x040C    22       RET
C:0x040D    5002     JNC      C:0411
C:0x040F    F7       MOV      @R1,A
C:0x0410    22       RET
C:0x0411    BBFE01   CJNE     R3,#OCRDL(0xFE),C:0415
C:0x0414    F3       MOVX     @R1,A
C:0x0415    22       RET
                C?PLDIIDATA:
C:0x0416    FA       MOV      R2,A
C:0x0417    E6       MOV      A,@R0
C:0x0418    FB       MOV      R3,A
C:0x0419    08       INC      R0
C:0x041A    08       INC      R0
C:0x041B    E6       MOV      A,@R0
C:0x041C    F9       MOV      R1,A
C:0x041D    25F0     ADD      A,B(0xF0)
C:0x041F    F6       MOV      @R0,A
C:0x0420    18       DEC      R0
C:0x0421    E6       MOV      A,@R0
C:0x0422    CA       XCH      A,R2
C:0x0423    3A       ADDC     A,R2
C:0x0424    F6       MOV      @R0,A
C:0x0425    22       RET
                C?CCASE:
C:0x0426    D083     POP      DPH(0x83)
C:0x0428    D082     POP      DPL(0x82)
C:0x042A    F8       MOV      R0,A
C:0x042B    E4       CLR      A
C:0x042C    93       MOVC     A,@A+DPTR
C:0x042D    7012     JNZ      C:0441
C:0x042F    7401     MOV      A,#0x01
C:0x0431    93       MOVC     A,@A+DPTR
C:0x0432    700D     JNZ      C:0441
C:0x0434    A3       INC      DPTR
C:0x0435    A3       INC      DPTR
C:0x0436    93       MOVC     A,@A+DPTR
C:0x0437    F8       MOV      R0,A
C:0x0438    7401     MOV      A,#0x01
C:0x043A    93       MOVC     A,@A+DPTR
C:0x043B    C0E0     PUSH     ACC(0xE0)
```

```
C:0x043D    E8       MOV      A,R0
C:0x043E    C0E0     PUSH     ACC(0xE0)
C:0x0440    22       RET
C:0x0441    7402     MOV      A,#0x02
C:0x0443    93       MOVC     A,@A+DPTR
C:0x0444    68       XRL      A,R0
C:0x0445    60EF     JZ       C:0436
C:0x0447    A3       INC      DPTR
C:0x0448    A3       INC      DPTR
C:0x0449    A3       INC      DPTR
C:0x044A    80DF     SJMP     C:042B
     6: void main()
     7: {
     8:
     9:
    10:
    11:   EA = 0;              // estetään keskeytykset
C:0x044C    C2AF     CLR      EA(0xA8.7)
    12:   TR0 = 0;             // pysäytetään ajastin 0
C:0x044E    C28C     CLR      TR0(0x88.4)
    13:   TMOD &= ~0x0F;       // tyhjennetään ajastin 0:n moodibitit
C:0x0450    5389F0   ANL      TMOD(0x89),#B(0xF0)
    14:   TMOD |= 0x01;        // asetetaan ajastin 0 16-bittiseksi ilman esijakajaa
C:0x0453    438901   ORL      TMOD(0x89),#0x01
    15:   PT0 = 0;             // asetetaan matala prioriteetti ajastin 0:lle
C:0x0456    C2B9     CLR      PT0(0xB8.1)
    16:   ET0 = 1;             // sallitaan ajastin 0 -keskeytys
C:0x0458    D2A9     SETB     ET0(0xA8.1)
    17:   TR0 = 1;             // käynnistetään ajastin 0
C:0x045A    D28C     SETB     TR0(0x88.4)
    18:   EA = 1;
C:0x045C    D2AF     SETB     EA(0xA8.7)
    19:   P2M1 = 0;
C:0x045E    E4       CLR      A
C:0x045F    F5A4     MOV      P2M1(0xA4),A
    20:   P1M1 = 0;
    21:
C:0x0461    F591     MOV      P1M1(0x91),A
    22:   SCON   = 0x52;        // UART-piirin asetukset
C:0x0463    759852   MOV      SCON(0x98),#0x52
    23:   BRGR0  = 0xF0;        // 9600 bit/s, 8 bittiä, ei pariteettibittiä, 1 loppubitti */
C:0x0466    75BEF0   MOV      BRGR0(0xBE),#B(0xF0)
    24:   BRGR0  = 0xF0;
C:0x0469    75BEF0   MOV      BRGR0(0xBE),#B(0xF0)
    25:   BRGR1  = 0x02;
C:0x046C    75BF02   MOV      BRGR1(0xBF),#0x02
    26:   BRGCON = 0x03;
C:0x046F    75BD03   MOV      BRGCON(0xBD),#0x03
    27:   P2 = 0x00;
    28:
C:0x0472    F5A0     MOV      P2(0xA0),A
    29:   while (1)
    30:   {
    31:     P2 ^= 0xFF;
C:0x0474    63A0FF   XRL      P2(0xA0),#OCRDH(0xFF)
    32:     if (timer0>=40 || a % 40 ==0)
C:0x0477    C3       CLR      C
C:0x0478    E51D     MOV      A,0x1D
C:0x047A    9428     SUBB     A,#0x28
C:0x047C    E51C     MOV      A,0x1C
C:0x047E    9400     SUBB     A,#0x00
C:0x0480    500E     JNC      C:0490
C:0x0482    AE1E     MOV      R6,0x1E
C:0x0484    AF1F     MOV      R7,0x1F
C:0x0486    7C00     MOV      R4,#0x00
C:0x0488    7D28     MOV      R5,#0x28
C:0x048A    7169     ACALL    C?UIDIV(C:0369)
C:0x048C    ED       MOV      A,R5
C:0x048D    4C       ORL      A,R4
C:0x048E    70E4     JNZ      C:0474
    33:                printf ("Tämä ohjelma kirjoittaa\nsarjaporttiin sarjassa\n\n");
```

```
C:0x0490    7BFF     MOV      R3,#OCRDH(0xFF)
C:0x0492    7A04     MOV      R2,#0x04
C:0x0494    79D8     MOV      R1,#I2CON(0xD8)
C:0x0496    116F     ACALL    PRINTF(C:006F)
    35:   }
    36: }
    37: //*****************************************************************************
    38: //static void timer0_isr (void);
    39: //Tällä funktiolla luodaan keskeytysrutiini ajastin 0:lle. Funktiota ei kutsuta koskaan
    40: //C-ohjelmalla, vaan se suoritetaan automaattisesti ajastin 0:n ylivuodon tapahtuessa.
    41: //*****************************************************************************
C:0x0498    80DA     SJMP     C:0474
    42: static void timer0_isr (void) interrupt 1 using 1
    43: {
    44:
C:0x049A    C0E0     PUSH     ACC(0xE0)
C:0x049C    C0D0     PUSH     PSW(0xD0)
    45:  TR0 = 0;        // pysäytetään ajastin 0
C:0x049E    C28C     CLR      TR0(0x88.4)
    46:  TR0 = 1;        // käynnistetään ajastin 0 uudelleen
C:0x04A0    D28C     SETB     TR0(0x88.4)
    47:  if (timer0>40 || a>40) // mikäli keskeytys on tapahtunut yli 40 kertaa,
C:0x04A2    D3       SETB     C
C:0x04A3    E51D     MOV      A,0x1D
C:0x04A5    9428     SUBB     A,#0x28
C:0x04A7    E51C     MOV      A,0x1C
C:0x04A9    9400     SUBB     A,#0x00
C:0x04AB    500A     JNC      C:04B7
C:0x04AD    E51F     MOV      A,0x1F
C:0x04AF    9428     SUBB     A,#0x28
C:0x04B1    E51E     MOV      A,0x1E
C:0x04B3    9400     SUBB     A,#0x00
C:0x04B5    400C     JC       C:04C3
    48:  {                           // nollataan keskeytyslaskuri
    49:         timer0=0;
C:0x04B7    751C00   MOV      0x1C,#0x00
C:0x04BA    751D00   MOV      0x1D,#0x00
    50:         a=0;
C:0x04BD    751E00   MOV      0x1E,#0x00
C:0x04C0    751F00   MOV      0x1F,#0x00
    51:  }
    52:         timer0++;              //kasvatetaan laskurin arvoa yhdellä
C:0x04C3    051D     INC      0x1D
C:0x04C5    E51D     MOV      A,0x1D
C:0x04C7    7002     JNZ      C:04CB
C:0x04C9    051C     INC      0x1C
    53:         a++;             //kasvatetaan apumuuttujan arvoa yhdellä
C:0x04CB    051F     INC      0x1F
C:0x04CD    E51F     MOV      A,0x1F
C:0x04CF    7002     JNZ      C:04D3
C:0x04D1    051E     INC      0x1E
    54: }
C:0x04D3    D0D0     POP      PSW(0xD0)
C:0x04D5    D0E0     POP      ACC(0xE0)
C:0x04D7    32       RETI
C:0x04D8    54E4     ANL      A,#FMCON(0xE4)
C:0x04DA    6D       XRL      A,R5
C:0x04DB    E4       CLR      A
C:0x04DC    206F68   JB       0x2D.7,C:0547
C:0x04DF    6A       XRL      A,R2
C:0x04E0    656C     XRL      A,0x6C
C:0x04E2    6D       XRL      A,R5
C:0x04E3    6120     AJMP     C:0320
C:0x04E5    6B       XRL      A,R3
C:0x04E6    69       XRL      A,R1
C:0x04E7    726A     ORL      C,0x2D.2
C:0x04E9    6F       XRL      A,R7
C:0x04EA    69       XRL      A,R1
C:0x04EB    7474     MOV      A,#0x74
C:0x04ED    6161     AJMP     C:0361
C:0x04EF    0A       INC      R2
```

```
C:0x04F0    73       JMP      @A+DPTR
C:0x04F1    6172     AJMP     C:0372
C:0x04F3    6A       XRL      A,R2
C:0x04F4    6170     AJMP     C:0370
C:0x04F6    6F       XRL      A,R7
C:0x04F7    7274     ORL      C,0x2E.4
C:0x04F9    7469     MOV      A,#0x69
C:0x04FB    69       XRL      A,R1
C:0x04FC    6E       XRL      A,R6
C:0x04FD    207361   JB       0x2E.3,C:0561
C:0x0500    726A     ORL      C,0x2D.2
C:0x0502    6173     AJMP     C:0373
C:0x0504    73       JMP      @A+DPTR
C:0x0505    610A     AJMP     C:030A
C:0x0507    0A       INC      R2
C:0x0508    00       NOP
                PUTCHAR:
C:0x0509    EF       MOV      A,R7
C:0x050A    B40A07   CJNE     A,#0x0A,C:0514
C:0x050D    740D     MOV      A,#0x0D
C:0x050F    120514   LCALL    C:0514
C:0x0512    740A     MOV      A,#0x0A
C:0x0514    309811   JNB      RI(0x98.0),C:0528
C:0x0517    A899     MOV      R0,SBUF(0x99)
C:0x0519    B8130C   CJNE     R0,#0x13,C:0528
C:0x051C    C298     CLR      RI(0x98.0)
C:0x051E    3098FD   JNB      RI(0x98.0),C:051E
C:0x0521    A899     MOV      R0,SBUF(0x99)
C:0x0523    C298     CLR      RI(0x98.0)
C:0x0525    B811F6   CJNE     R0,#0x11,C:051E
C:0x0528    3099FD   JNB      TI(0x98.1),C:0528
C:0x052B    C299     CLR      TI(0x98.1)
C:0x052D    F599     MOV      SBUF(0x99),A
C:0x052F    22       RET
   258:              MOV     R0,#IDATALEN - 1
C:0x0530    78FF     MOV      R0,#OCRDH(0xFF)
   259:              CLR     A
C:0x0532    E4       CLR      A
   260: IDATALOOP:   MOV     @R0,A
C:0x0533    F6       MOV      @R0,A
   261:              DJNZ    R0,IDATALOOP
C:0x0534    D8FD     DJNZ     R0,IDATALOOP(C:0533)
   299:              MOV     SP,#?STACK-1
C:0x0536    758121   MOV      SP(0x81),#0x21
   300:              JMP     ?C_START
C:0x0539    02044C   LJMP     main(C:044C)
```