KANDULA SRINIVASA REDDY MEMORIAL COLLEGE OF ENGINEERING
(AUTONOMOUS)

KADAPA-516003. AP

(Approved by AICTE, Affiliated to JNTU A, Ananthapuramu, Accredited by NAAC)

(An ISO 9001-2008 Certified Institution)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Certification Course

On

"MSP 430 Launch Pad Programming"

Resource Persons : Dr.G.Hemalatha, Professor Dept. of ECE, KSRMCE
Smt.K.Divyalakshmi, Assistant professor, Dept. of ECE, KSRMCE

Course Coordinators : Dr.G.Hemalatha, Professor Dept. of ECE, KSRMCE
Smt.K.Divyalakshmi, Assistant professor, Dept. of ECE, KSRMCE

Duration : 04/02/2019 to 23/02/2019

K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India– 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

Lr./KSRMCE/ECE/2018-19/ Date:30/01/2019

To
The Principal,
KSRMCE,
Kadapa.

Respected Sir,

Sub: Permission to Conduct Certification Course on "**MSP430 Launchpad programming**" **04/02/2019 to 23/02/2019**–Req- Reg.

The Department of Electronics and communication engineering is planning to offer a Certification Course on "MSP430 Launchpad programming" to B. Tech. students. The course will be conducted from **04/02/2019 to 23/02/2019**. In this regard, I kindly request you to grant permission to conduct a Certification Course.

Thanking you sir,

K Divyalakshmi
Yours faithfully

(Smt.K.Divyalakshmi, Asst.Professor in ECED)

forwarded to the Principal sir

V.S.S. Murthy

PRINCIPAL
K.S.R.M. COLLEGE OF ENGINEERING
KADAPA-516005, (A.P.)

K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India– 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

Cr./KSRMCE/ECE/2018-19/ Date: 31/01/2019

Circular

The Department of Electronics and Communication Engineering is offering a Certification Course on "MSP 430 Launchpad programming" from **04/02/2019 to 23/02/2019** to B.Tech students. In this regard, interested students are requested to register their names for the Certification Course with Course Coordinator._

For further information contact the Course Coordinator.

Course Coordinator: Smt.K.Divyalakshmi, Asst.professor, Dept. of ECE.-KSRMCE.
Contact No: 9494947993

HoD

Dept. of ECE

Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003

Cc to:

IQAC-KSRMCE

K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India– 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

Date:

DEPARTMENT OF ECE

REGISTRATION FORM

Certification Course
On
"MSP 430 Launchpad programming" From 04/02/2019 to 23/02/2019

S.No	Full Name	Roll Number	Branch	Semester	Signature

Coordinators **HoD**

K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India– 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

DEPARTMENT OF ECE

REGISTRATION FORM

Certification Course
On
"MSP 430 Launch pad programming" From 04/02/2019 to 23/02/2019

S.No	Full Name	Roll Number	Branch	Semester	Signature
1	Barigela Udaykumar	169Y1A0410	ECE	VI SEM	
2	Barinela Suresh	169Y1A0411	ECE	VI SEM	
3	Basireddy Rajasekhar	169Y1A0412	ECE	VI SEM	RajSekhar
4	Batthala Chandanapriya	169Y1A0413	ECE	VI SEM	Chandanapriya
5	Bayyapu Reddy Sushma	169Y1A0414	ECE	VI SEM	Sushma
6	Bommireddy Navyasree	169Y1A0415	ECE	VI SEM	NavyaSree
7	Cheppali Prudhvi Raju	169Y1A0416	ECE	VI SEM	Prudhviraj
8	Cherukuri Sashitha	169Y1A0417	ECE	VI SEM	Sashitha
9	Chilekampalle Sai Sandeep Reddy	169Y1A0418	ECE	VI SEM	Sandeep
10	Chintakunta Veera Manisha	169Y1A0419	ECE	VI SEM	C. Manisha
11	Chinthakunta Venkata Manirathnam	169Y1A0420	ECE	VI SEM	
12	Chowdam Sudarshan	169Y1A0421	ECE	VI SEM	Sudarshan
13	Chuppala Nararajasekhar	169Y1A0422	ECE	VI SEM	Nararajasekhar
14	D Vishnu	169Y1A0423	ECE	VI SEM	Vishnu
15	Dalavapalli Aravind	169Y1A0424	ECE	VI SEM	Aravind
16	Dalavaye Bhaskar	169Y1A0425	ECE	VI SEM	Bhaskar
17	Davanam Deepa	169Y1A0426	ECE	VI SEM	Deepa
18	Edde Rajasekhar Reddy	169Y1A0427	ECE	VI SEM	Rajasekhar
19	Eragamreddy Siva Reddy	169Y1A0428	ECE	VI SEM	Siva Reddy
20	Gaddam Rachana	169Y1A0429	ECE	VI SEM	Rachana
21	Gajjala Deepthi	169Y1A0430	ECE	VI SEM	Deepthi
22	Galla Manisha	169Y1A0431	ECE	VI SEM	Manisha
23	Gondipalle Anil Kumar	169Y1A0432	ECE	VI SEM	Anil Kumar
24	Gontimukkala Venkata Kalyan	169Y1A0433	ECE	VI SEM	Kalyan
25	Gosula Sreesai	169Y1A0434	ECE	VI SEM	
26	Goturu Anusha	169Y1A0435	ECE	VI SEM	Anusha
27	Gundarapu Kishore	169Y1A0436	ECE	VI SEM	Kishore

28	Gundluri Vinitha	169Y1A0437	ECE	VI SEM	Vinitha
29	Jangidi Pallavi	169Y1A0439	ECE	VI SEM	Pallavi
30	Kadiri Krishna Moorthy Reddy	169Y1A0440	ECE	VI SEM	Krishna
31	Kalyandurg Bhanuprakash	169Y1A0441	ECE	VI SEM	Bhanu
32	Kamalapuram Hariprasad Goud	169Y1A0442	ECE	VI SEM	Haripas
33	Kandlakuti Sumalatha	169Y1A0443	ECE	VI SEM	Sumal
34	Kataboina Gurrappa	169Y1A0444	ECE	VI SEM	
35	Kathi Chandana	169Y1A0445	ECE	VI SEM	Chand

K. Divyalakshmi.
Coordinators

HoD

Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003

Syllabus of Certification Course

Course name : MSP 430 launchpad programming

Course objectives:

1. This course will introduce you to the MSP430 and embedded software in general.
2. You will also learn how to implement a basic task scheduler. From setting up ports and registers, to more advanced subjects like callback functions, structs, and timers,
3. you will learn how to program an MSP430 to do precisely timed tasking in a fairly simple manner.

Course outcomes:

After studying this course, students will be able to:

1. Understand the architectural features and instruction set of 16 bit microcontroller MSP430.
2. Develop programs using the various instructions of MSP430 for different applications.
3. Understand the functions of the various peripherals which are interfaced with MSP430 microcontroller.
4. Describe the power saving modes in MSP430.
5. Explain the low power applications using the MSP430 microcontroller.

UNIT-I

MSP430 Architecture:

Introduction –Where does the MSP430 fit, the outside view, The inside view-Functional block diagram, Memory, Central Processing Unit, Memory Mapped Input and Output, Clock Generator, Exceptions: Interrupts and Resets, MSP430 family.

UNIT-II

Addressing Modes & Instruction Set-

Addressing Modes, Instruction set, Constant Generator and Emulated Instructions, Program Examples. Module-3 Clock System, Interrupts and Operating Modes.

UNIT-III

Clock System, Interrupts and Operating Modes-

Clock System, Interrupts, What happens when an interrupted is requested, Interrupt Service Routines, Low Power Modes of Operation, Watchdog Timer, Basic Timer1, Real Time Clock, Timer-A: Timer Block, Capture/Compare Channels, Interrupts from Timer-A.

UNIT-IV

Analog Input-Output and PWM -

Comparator-A, ADC10, ADC12, Sigma-Delta ADC, Internal Operational Amplifiers, DAC, Edge Aligned PWM, Simple PWM, Design of PWM. LCD interfacing.

UNIT-V

Digital Input-Output and Serial Communication:

Parallel Ports, Lighting LEDs, Flashing LEDs, Read Input from a Switch, Toggle the LED state by pressing the push button, LCD interfacing. Asynchronous Serial Communication, Asynchronous Communication with USCI_A, Communications, Peripherals in MSP430, Serial Peripheral Interface.

Textbooks/Reference books:

1.MSP430 LaunchPad Programming Kindle Edition by “Agus Kurniawan”

2.Embedded Systems Design using the MSP430FR2355 LaunchPad by “Brock J. LaMeres”

3.Programmable Microcontrollers with Applications: Msp430 Launchpad with CCS and Grace book review by “Cem unsalan”

Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003.

K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India– 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

SCHEDULE

Department of ECE

Certification Course
On
"MSP 430 Launchpad programming" From 04/02/2019 to 23/02/2019

Date	Timing	Resource person	Topic to be covered
04-02-2019	3 PM to4PM	Dr.G.Hemalatha	**MSP430 Architecture:** Introduction –Where does the MSP430 fit
04-02-2019	4PM to 5PM	Dr.G.Hemalatha	The outside view
05-02-2019	4PM to 5PM	Smt.K.Divyalakshmi	The inside view
06-02-2019	3PM to 4PM	Smt.K.Divyalakshmi	Functional block diagram,
06-02-2019	4PM to 5PM	Dr.G.Hemalatha	Memory, Central Processing Unit.
07-02-2019	4PM to 5PM	Dr.G.Hemalatha	Memory Mapped Input and Output
08-02-2019	3PM to 4PM	Smt.K.Divyalakshmi	Clock Generator, Exceptions
08-02-2019	4PM to 5PM	Smt.K.Divyalakshmi	Interrupts and Resets, MSP430 family.
11-02-2019	3PM to 4PM	Dr.G.Hemalatha	**Addressing Modes & Instruction Set:** Addressing Modes,
11-02-2019	4PM to 5PM	Dr.G.Hemalatha	Instruction set,
12-02-2019	2PM to 4PM	Smt.K.Divyalakshmi	Constant Generator and Emulated Instructions
12-02-2019	4PM to 5PM	Smt.K.Divyalakshmi	Program Examples.
13-02-2019	2PM to 4PM	Dr.G.Hemalatha	Module-3Clock System,
13-02-2019	4PM to 5PM	Dr.G.Hemalatha	Interrupts and Operating Modes
14-02-2019	2PM to 4PM	Smt.K.Divyalakshmi	**Clock System, Interrupts and Operating Modes:** Clock System, Interrupts
14-02-2019	4PM to 5PM	Smt.K.Divyalakshmi	What happens when an interrupted is requested,
15-02-2019	2PM to 4PM	Dr.G.Hemalatha	Interrupt Service Routines,
15-02-2019	4PM to 5PM	Dr.G.Hemalatha	Low Power Modes of Operation, Watchdog Timer,
16-02-2019	2PM to 4PM	Smt.K.Divyalakshmi	Basic Timer1, Real Time Clock, Timer-A: Timer Block,
16-02-2019	4PM to 5PM	Smt.K.Divyalakshmi	Capture/Compare Channels, Interrupts from Timer-A
18-02-2019	2PM to 4PM	Dr.G.Hemalatha	**Analog Input-Output and PWM :**

			Comparator-A, ADC10, ADC12, Sigma-Delta ADC
18-02-2019	4PM to 5PM	Dr.G.Hemalatha	Internal Operational Amplifiers, DAC, Edge Aligned PWM,
19-02-2019	2PM to 4PM	Smt.K.Divyalakshmi	Simple PWM, Design of PWM. LCD interfacing.
19-02-2019	4PM to 5PM	Smt.K.Divyalakshmi	**Digital Input-Output and Serial Communication:** Parallel Ports, Lighting LEDs,
20-02-2019	2PM to 4PM	Dr.G.Hemalatha	Flashing LEDs, Read Input from a Switch.
20-02-2019	4PM to 5PM	Dr.G.Hemalatha	Toggle the LED state by pressing the push button
21-02-2019	2PM to 4PM	Smt.K.Divyalakshmi	LCD interfacing. Asynchronous Serial Communication,
21-02-2019	4PM to 5PM	Smt.K.Divyalakshmi	Asynchronous Communication with USCI_A,
22-02-2019	4PM to 5PM	Dr.G.Hemalatha	Communications, Peripherals in MSP430, Serial Peripheral Interface.
23-02-2019	4PM to 5PM	Dr.G.Hemalatha Smt.K.Divyalakshmi	Valedictory

Resource Person(s)

Coordinator(s)

HoD

Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003

K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa, Andhra Pradesh, India– 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

DEPARTMENT OF ECE

Attendance sheet of Certification Course on “MSP 430 Launchpad programming” From 04/02/2019 to 23/02/2019

S. No.	Roll No.	Name	04/02/2019	05/02/2019	06/02/2019	07/02/2019	08/02/2019	11/02/2019	12/02/2019	13/02/2019	14/02/2019	15/02/2019	16/02/2019	18/02/2019	19/02/2019	20/02/2019	21/02/2019	22/02/2019	23/02/2019
1	169Y1A0410	Barigela Udaykumar	P	A	P	A	P	A	A	A	A	A	A	A	A	A	A	A	A
2	169Y1A0411	Barinela Suresh	P	A	A	A	P	A	A	P	P	A	A	P	P	A	A	P	A
3	169Y1A0412	Basireddy Rajasekhar	A	A	A	A	A	A	P	P	A	A	A	P	P	P	P	A	A
4	169Y1A0413	Batthala Chandanapriya (W)	P	P	A	A	P	A	P	A	A	A	P	A	A	A	P	P	A
5	169Y1A0414	Bayyapu Reddy Sushma (W)	A	A	P	P	A	P	A	P	A	P	A	P	A	P	A	P	A
6	169Y1A0415	Bommireddy Navyasree (W)	P	A	A	A	A	A	A	A	A	A	A	A	A	A	P	P	A
7	169Y1A0416	Cheppali Prudhvi Raju	A	P	A	A	A	A	A	A	A	A	A	P	P	A	A	P	P
8	169Y1A0417	Cherukuri Sashitha (W)	P	P	A	A	A	A	A	A	A	A	A	A	A	P	P	P	A
9	169Y1A0418	Chilekampalle Sai Sandeep Reddy	A	A	P	P	A	A	A	A	A	A	P	P	P	A	A	A	A
10	169Y1A0419	Chintakunta Veera Manisha (W)	P	A	P	P	A	A	A	A	A	A	A	A	A	A	P	P	A
11	169Y1A0420	ChinthakuntaVe	A	P	P	P	P	A	A	A	A	A	A	A	A	A	A	P	P

		nkata Manirathnam																	
12	169Y1A0421	Chowdam Sudarshan	P	A	P	A	P	A	P	A	P	P	A	P	P	P	A	P	A
13	169Y1A0422	Chuppala Nararajasekhar	P	P	P	P	A	P	A	P	A	A	P	P	P	A	P	A	A
14	169Y1A0423	D Vishnu	A	P	A	P	P	P	P	P	P	P	P	P	P	P	P	P	A
15	169Y1A0424	Dalavapalli Aravind	P	A	P	A	A	P	P	A	P	P	P	P	P	P	A	P	A
16	169Y1A0425	Dalavaye Bhaskar	P	P	P	P	P	A	A	P	P	P	A	P	P	P	P	A	P
17	169Y1A0426	Davanam Deepa (W)	A	P	A	P	P	P	P	P	P	A	A	P	P	P	A	P	P
18	169Y1A0427	Edde Rajasekhar Reddy	P	P	P	A	A	P	P	A	A	A	P	A	A	P	P	P	A
19	169Y1A0428	Eragamreddy Siva Reddy	P	A	P	P	P	A	P	P	P	P	P	A	P	A	P	A	P
20	169Y1A0429	Gaddam Rachana (W)	A	P	A	P	A	P	A	P	P	P	P	P	A	P	A	A	P
21	169Y1A0430	Gajjala Deepthi (W)	P	P	P	A	P	P	P	P	P	P	P	P	P	P	A	A	P
22	169Y1A0431	Galla Manisha (W)	P	A	A	P	A	P	P	A	P	A	A	P	A	P	P	P	P
23	169Y1A0432	Gondipalle Anil Kumar	A	P	P	P	P	A	A	P	A	P	A	P	P	P	P	P	P
24	169Y1A0433	Gontimukkala Venkata Kalyan	P	P	P	A	P	P	P	P	P	P	P	P	P	P	P	A	P
25	169Y1A0434	Gosula Sreesai (W)	P	A	A	P	P	P	P	P	P	A	P	P	A	A	P	P	P
26	169Y1A0435	Goturu Anusha (W)	A	P	P	P	A	A	A	A	A	A	A	A	A	A	P	A	P
27	169Y1A0436	Gundarapu Kishore	P	P	A	A	P	P	P	P	P	P	A	A	P	P	P	P	P
28	169Y1A0437	Gundluri Vinitha (W)	A	A	P	P	P	P	P	P	P	P	P	P	P	P	A	P	P
29	169Y1A0439	Jangidi Pallavi (W)	P	P	A	P	A	A	P	P	A	A	P	P	A	P	A	P	P
30	169Y1A0440	Kadiri Krishna Moorthy Reddy	P	P	P	A	P	P	P	A	P	P	A	P	P	P	A	P	P
31	169Y1A0441	Kalyandurg Bhanuprakash	A	A	P	P	P	P	A	P	P	P	P	A	P	P	P	A	A
32	169Y1A0442	Kamalapuram Hariprasad	P	P	A	P	P	P	P	P	P	P	P	P	P	P	P	P	P

		Goud																	
33	169Y1A0443	Kandlakuti Sumalatha (W)	P	P	A	P	P	P	P	A	A	P	P	P	A	P	P	A	P
34	169Y1A0444	Kataboina Gurrappa	A	P	P	P	P	P	P	P	A	P	P	P	P	P	P	A	P
35	169Y1A0445	Kathi Chandana (W)	P	P	P	P	P	P	A	P	P	P	P	P	A	P	P	P	A

Coordinator(s)

HoD

Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003.

Photos :

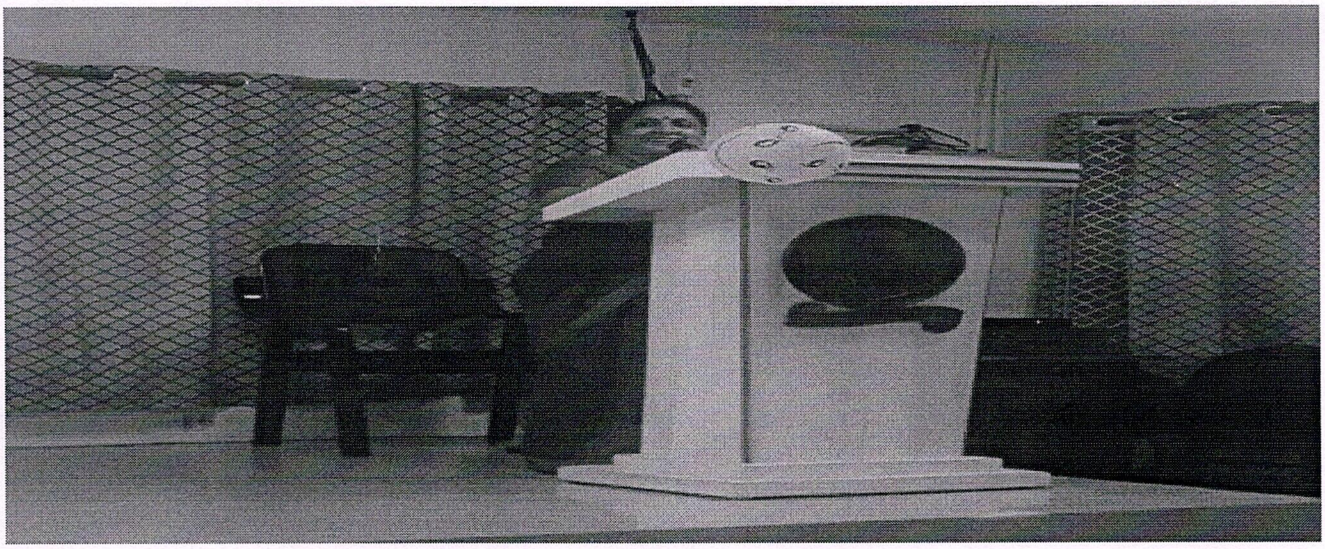


Fig : Resource person delivering the lecture

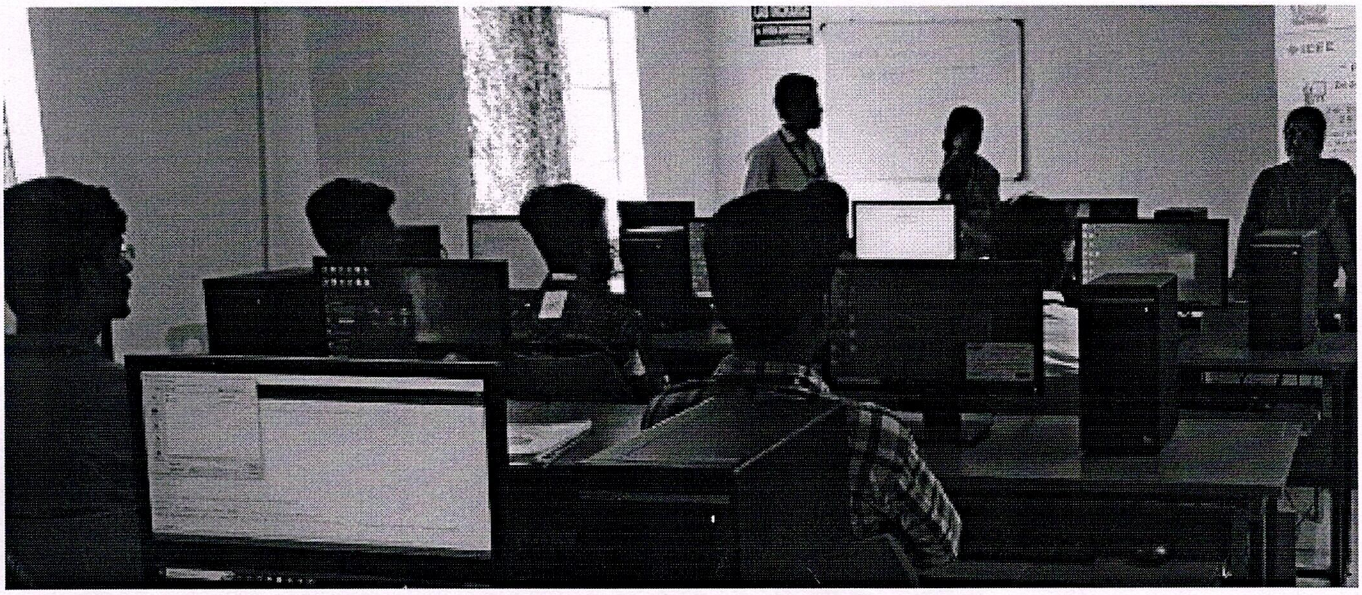


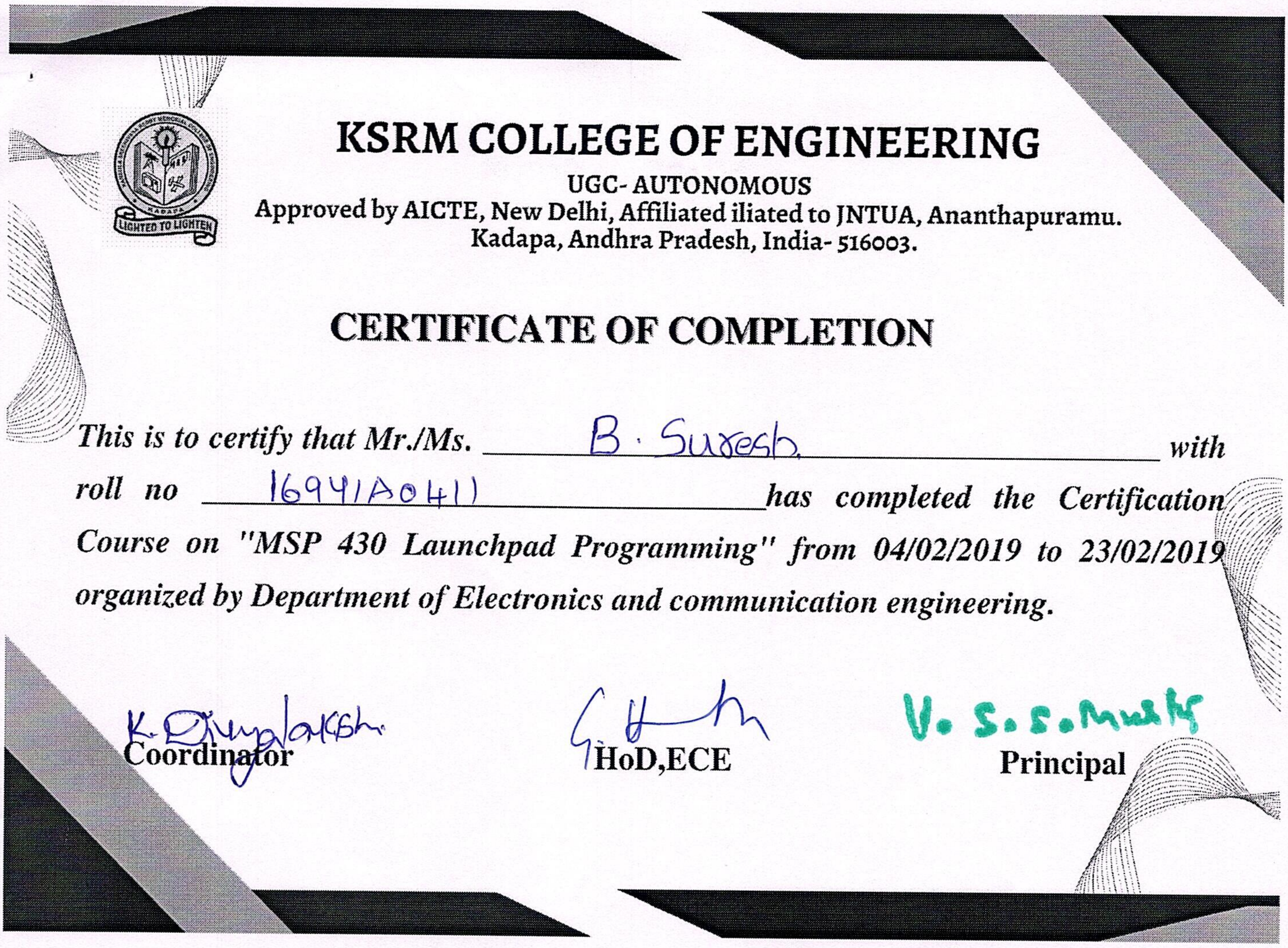
Fig : Students listening to the lecture.

K.Omprakash

Coordinators

HoD

Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003



KSRM COLLEGE OF ENGINEERING

UGC- AUTONOMOUS
Approved by AICTE, New Delhi, Affiliated iliated to JNTUA, Ananthapuramu.
Kadapa, Andhra Pradesh, India- 516003.

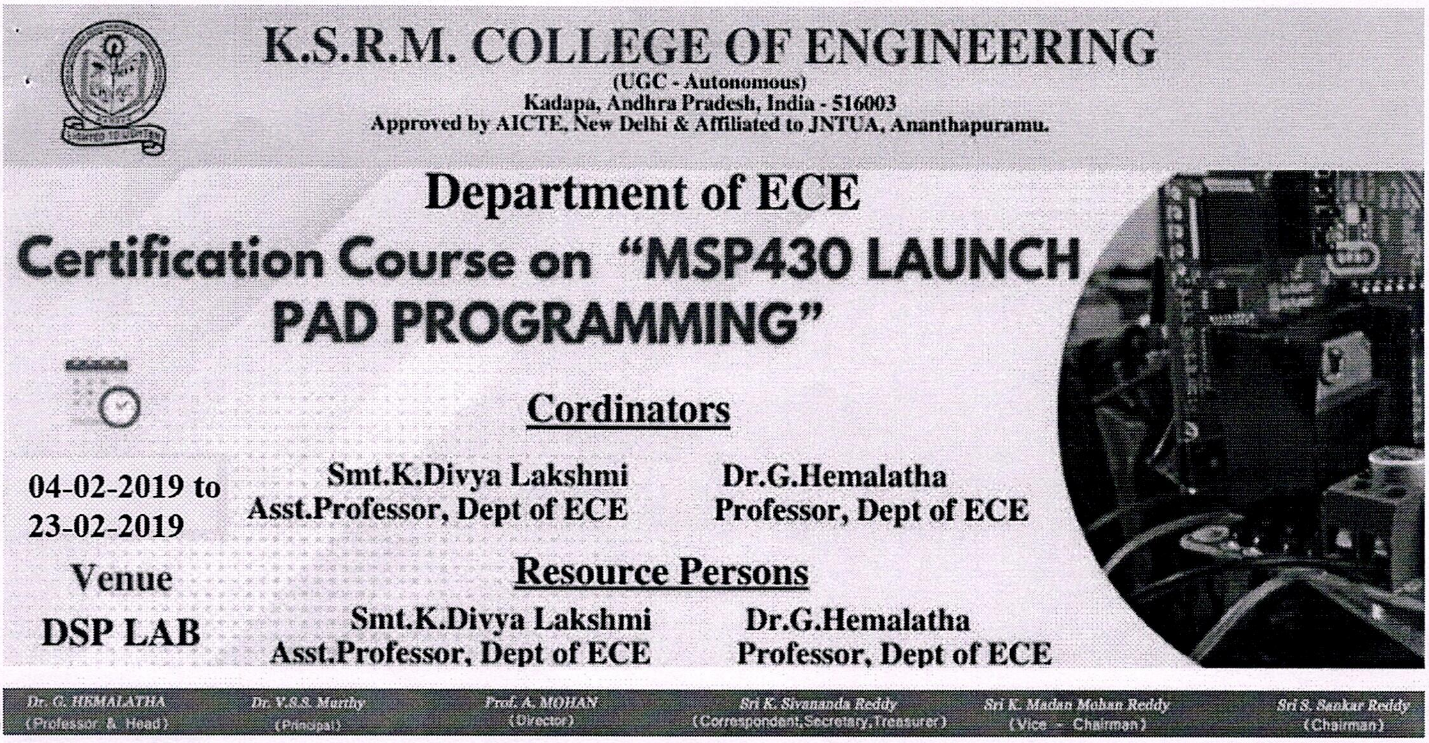
CERTIFICATE OF COMPLETION

This is to certify that Mr./Ms. B. Suresh *with roll no* 16941A0411 *has completed the Certification Course on "MSP 430 Launchpad Programming" from 04/02/2019 to 23/02/2019 organized by Department of Electronics and communication engineering.*

K. Divyalaksh.
Coordinator

HoD,ECE

V. S. S. Murthy
Principal



K.S.R.M. COLLEGE OF ENGINEERING

(UGC - Autonomous)
Kadapa, Andhra Pradesh, India - 516003
Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

Department of ECE

Certification Course on "MSP430 LAUNCH PAD PROGRAMMING"

04-02-2019 to 23-02-2019

Venue

DSP LAB

Cordinators

Smt.K.Divya Lakshmi
Asst.Professor, Dept of ECE

Dr.G.Hemalatha
Professor, Dept of ECE

Resource Persons

Smt.K.Divya Lakshmi
Asst.Professor, Dept of ECE

Dr.G.Hemalatha
Professor, Dept of ECE

Dr. G. HEMALATHA (Professor & Head)
Dr. V.S.S. Murthy (Principal)
Prof. A. MOHAN (Director)
Sri K. Sivananda Reddy (Correspondent,Secretary,Treasurer)
Sri K. Madan Mohan Reddy (Vice - Chairman)
Sri S. Sankar Reddy (Chairman)

K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)
Kadapa,Andhra Pradesh, India– 516 003
Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.
An ISO 14001:2004 & 9001: 2015 Certified Institution

ACTIVITY REPORT

Certification Course

On

"MSP 430 Launchpad Programming "

From 04th Feb 2019 to 23rd feb 2019

Target Group	:	**B.Tech Students**
Details of Participants	:	**35 Students**
Resource persons	:	**Dr.G.Hemalatha Prof, Dept. of ECE** **Smt .K.Divyalakshmi, Asst.Prof, Dept. of ECE**
Coordinators	:	**Dr.G.Hemalatha Prof, Dept. of ECE** **Smt .K.Divyalakshmi, Asst.Prof, Dept. of ECE**
Organizing Department	:	**Department of Electronics and Communication Engineering**
Venue	:	**DSP lab**

Description:

Certification Course on "MSP 430 Launchpad Programming" was organized by Dept. of ECE from 04/02/2019 to 23/02/2019 . Dr .G.Hemalatha, Smt.K.Divyalakshmi acted as Course instructor. MSP430 Architecture, Addressing Modes & Instruction Set, Clock System, Interrupts and Operating Modes, Analog Input-Output and PWM, Digital Input-Output and Serial Communication was explained.

KSRM COLLEGE OF ENGINEERING

UGC- AUTONOMOUS
Approved by AICTE, New Delhi, Affiliated iliated to JNTUA, Ananthapuramu.
Kadapa, Andhra Pradesh, India- 516003.

CERTIFICATE OF COMPLETION

This is to certify that Mr./Ms. G. Sreesai *with roll no* 169Y1A0434 *has completed the Certification Course on "MSP 430 Launchpad Programming" from 04/02/2019 to 23/02/2019 organized by Department of Electronics and communication engineering.*

K. Divyalakshmi
Coordinator

HoD,ECE

V. S. S. Murthy
Principal

KSRM COLLEGE OF ENGINEERING

UGC- AUTONOMOUS
Approved by AICTE, New Delhi, Affiliated iliated to JNTUA, Ananthapuramu.
Kadapa, Andhra Pradesh, India- 516003.

CERTIFICATE OF COMPLETION

This is to certify that Mr./Ms. G. Deepthi *with roll no* 169Y1A0430 *has completed the Certification Course on "MSP 430 Launchpad Programming" from 04/02/2019 to 23/02/2019 organized by Department of Electronics and communication engineering.*

Coordinator

HoD,ECE

V. S. S. Murthy
Principal

K.S.R.M. COLLEGE OF ENGINEERING

UGC - AUTONOMOUS

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.
Kadapa, Andhra Pradesh, India– 516 003

FEEDBACK FORM

Certification Course on "*MSP 430 Launchpad programming*", from 04-02-2019 to 23-02-2019

Organized

by

Department of Electronics & Communication Engineering

NAME: 16941A0418

Roll No:

S.No	Feedback Item	Excellent	Very Good	Good	Average	Below Average
1	Organization of certificate course and session planning by the instructor.					
2	Clarity in content delivery.					
3	Content is relevant and useful					
4	Adequate opportunity to interact with trainer					
5	Judicious mix of concepts. Principles and practices.					
6	Assignments and tasks are interesting and challenging.					
7	Overall rating					

Any suggestions for improvement.

Signature

K.S.R.M. COLLEGE OF ENGINEERING

(UGC-AUTONOMOUS)

Kadapa,Andhra Pradesh, India– 516 003

Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu.

An ISO 14001:2004 & 9001: 2015 Certified Institution

Certification Course on

“MSP 430 Launch pad programming”

04/02/2019 to 23/02/2019

Feedback responses

S.No.	Roll No	Year & Semester	Branch	Is the course content met your expectation	Is the lecture sequence well planned	The contents of the course is explained with examples	Is the level of course high	Is the course exposed you to the new knowledge and practices	Is the lecturer clear and easy to understand	Rate the value of course in increasing your skills	Any issues
1	169Y1A0410	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	3	5	Nothing
2	169Y1A0411	B.Tech VIsem	ECE	Yes	Yes	agree	Agree	Strongly agree	3	4	very good
3	169Y1A0412	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	5	5	Good
4	169Y1A0413	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	5	4	very good
5	169Y1A0414	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	4	5	Nothing
6	169Y1A0415	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	4	3	Good
7	169Y1A0416	B.Tech VIsem	ECE	Yes	Yes	agree	Agree	Strongly agree	5	4	Good
8	169Y1A0417	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	5	4	Nothing
9	169Y1A0418	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	5	5	Nothing

10	169Y1A0419	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	4	3	Very Good
11	169Y1A0420	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	4	5	Good
12	169Y1A0421	B.Tech VIsem	ECE	Yes	Yes	agree	Agree	Strongly agree	5	4	Good
13	169Y1A0422	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	3	5	Nothing
14	169Y1A0423	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	5	5	very good
15	169Y1A0424	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	5	4	Nothing
16	169Y1A0425	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	4	5	very good
17	169Y1A0426	B.Tech VIsem	ECE	Yes	Yes	agree	Agree	Strongly agree	5	3	No
18	169Y1A0427	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	3	4	Nothing
19	169Y1A0428	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	5	5	Good
20	169Y1A0429	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	3	3	Good
21	169Y1A0430	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	5	5	Good
22	169Y1A0431	B.Tech VIsem	ECE	Yes	Yes	agree	Agree	Strongly agree	5	5	Nothing
23	169Y1A0432	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	4	4	Good
24	169Y1A0433	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	4	4	Good
25	169Y1A0434	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	4	4	Good
26	169Y1A0435	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	5	5	Nothing
27	169Y1A0436	B.Tech VIsem	ECE	Yes	Yes	agree	Agree	Strongly agree	5	3	No
28	169Y1A0437	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	4	4	No
29	169Y1A0439	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	4	3	No
30	169Y1A0440	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	5	4	No

31	169Y1A0441	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	5	4	nothing
32	169Y1A0442	B.Tech VIsem	ECE	Yes	Yes	agree	Agree	Strongly agree	5	3	Nothing
33	169Y1A0443	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	4	4	No
34	169Y1A0444	B.Tech VI sem	ECE	Yes	Yes	agree	Agree	Strongly agree	4	5	Nothing
35	169Y1A0445	B.Tech VI sem	ECE	Yes	Yes	Strongly agree	Agree	Strongly agree	5	5	Good

K. Divyalakshmi
Coordinator

HoD

Professor & H.O.D.
Department of E.C.E.
K.S.R.M. College of Engineering
KADAPA - 516 003

MSP 430

Memory

Volatile Memory:

- Loses its contents when power is removed. It is usually called *random-access memory* or RAM.
- The vital feature is that data can be read or written with equal ease. Volatile memory is used for data, and small microcontrollers often have very little RAM, sometimes only a few tens of bytes.

Static RAM:

- means that it retains its data even if the clock is stopped (provided that power is maintained, of course).
- A single cell of static RAM needs six transistors.
- RAM therefore takes up a large area of silicon, which makes it expensive.

Dynamic RAM:

- This needs only one transistor per cell but must be refreshed regularly to maintain its contents, so it is not used in small microcontrollers.
- Most memory in a desktop computer is *dynamic* RAM.

Nonvolatile Memory:

- Retains its contents when power is removed and is therefore used for the program and constant data. It is usually called *read-only memory* or ROM

There are many types of nonvolatile memory in use:

- **Masked ROM:**
- The data are encoded into one of the masks used for photolithography and written into the IC during manufacture. This memory really is read-only.

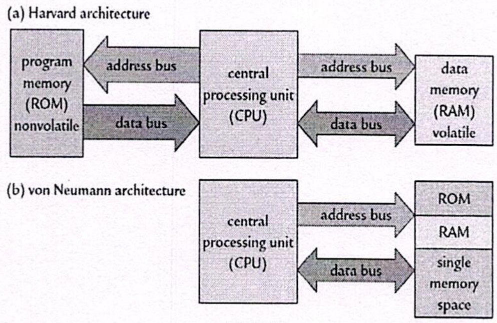
- It is used for the high-volume production of stable products, because any change to the data requires a new mask to be produced at great expense.
- Some MSP430 devices can be ordered with ROM, shown by a *C* in their part number. An example is the MSP430CG4619.
- **EPROM (electrically programmable ROM):**
- As its name implies, it can be programmed electrically but not erased.
- Devices must be exposed to ultraviolet (UV) light for about ten minutes to erase them.
- erasable devices need special packages with quartz windows, which are expensive.

Flash memory:

- This can be both programmed and erased electrically and is now by far the most common type of memory.
- The practical difference is that individual bytes of EEPROM can be erased but flash can be erased only in blocks.
- Most MSP430 devices use flash memory, shown by an *F* in the part number.
- Microcontrollers use **NOR flash**, which is slower to write but permits random access. **NAND flash** is used in bulk storage devices and can be accessed only serially in rows.

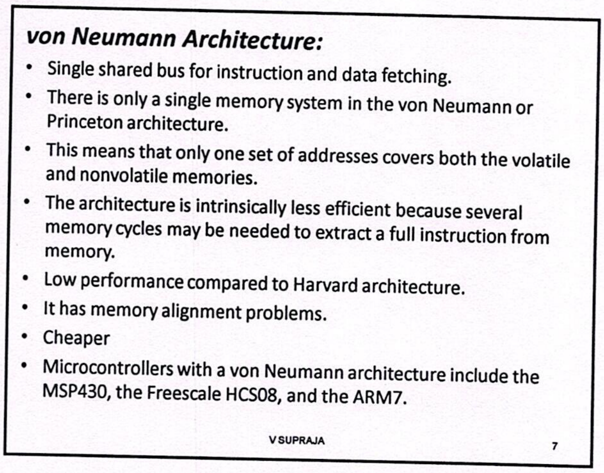
Harvard and Von-Neuman Architectures

Block Diagram:


(a) Harvard architecture
program memory (ROM) nonvolatile
address bus
data bus
central processing unit (CPU)
address bus
data bus
data memory (RAM) volatile
(b) von Neumann architecture
central processing unit (CPU)
address bus
data bus
ROM
RAM
single memory space

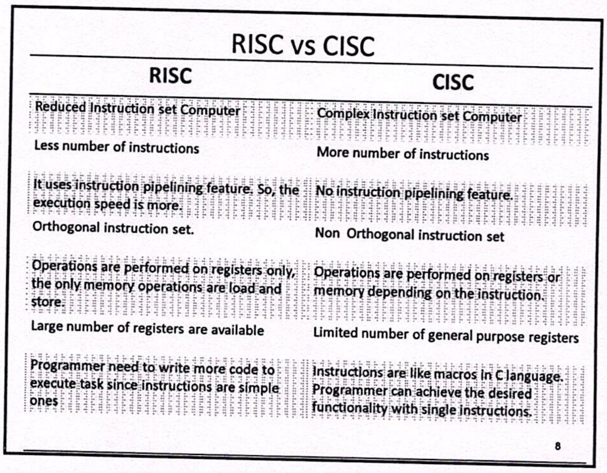
Harvard Architecture:

- The volatile (data) and nonvolatile (program) memories are treated as separate systems.
- Each with its own address and data bus. Many microcontrollers use this architecture, including Microchip PICs, the Intel 8051 and descendents, and the ARM9.
- The principal advantage is efficiency.
- It allows simultaneous access to the program and data memories. For instance, the CPU can read an operand from the data memory at the same time as it reads the next instruction from the program memory.
- Easier to pipeline so high performance can be achieved.
- No memory alignment problems.
- High Cost



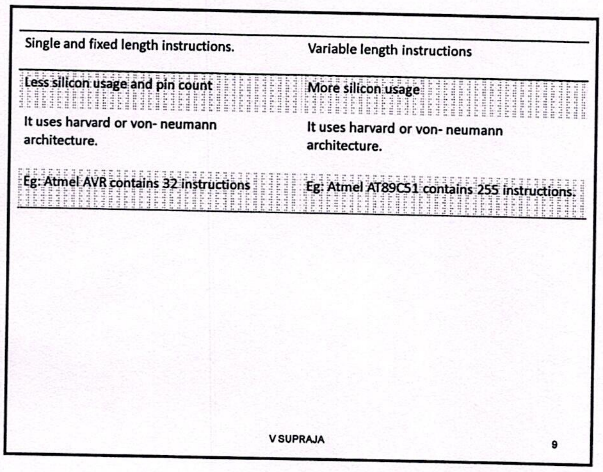
von Neumann Architecture:

- Single shared bus for instruction and data fetching.
- There is only a single memory system in the von Neumann or Princeton architecture.
- This means that only one set of addresses covers both the volatile and nonvolatile memories.
- The architecture is intrinsically less efficient because several memory cycles may be needed to extract a full instruction from memory.
- Low performance compared to Harvard architecture.
- It has memory alignment problems.
- Cheaper
- Microcontrollers with a von Neumann architecture include the MSP430, the Freescale HCS08, and the ARM7.

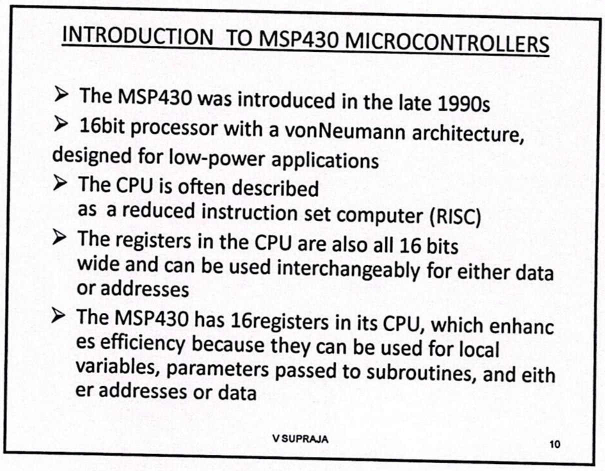


RISC vs CISC

RISC	CISC
Reduced Instruction set Computer	Complex Instruction set Computer
Less number of instructions	More number of instructions
It uses instruction pipelining feature. So, the execution speed is more.	No instruction pipelining feature.
Orthogonal instruction set.	Non Orthogonal instruction set
Operations are performed on registers only, the only memory operations are load and store.	Operations are performed on registers or memory depending on the instruction.
Large number of registers are available	Limited number of general purpose registers
Programmer need to write more code to execute task since instructions are simple ones	Instructions are like macros in C language. Programmer can achieve the desired functionality with single instructions.

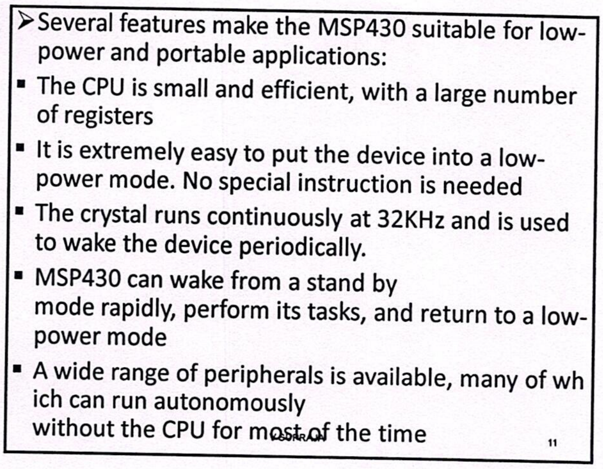


Single and fixed length instructions.	Variable length instructions
Less silicon usage and pin count	More silicon usage
It uses harvard or von- neumann architecture.	It uses harvard or von- neumann architecture.
Eg: Atmel AVR contains 32 instructions	Eg: Atmel AT89C51 contains 255 instructions.



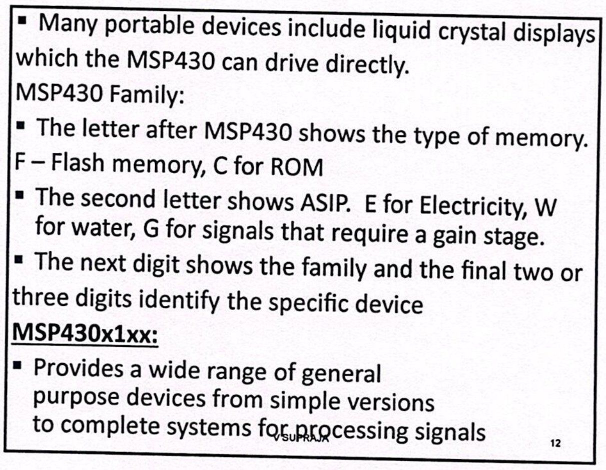
INTRODUCTION TO MSP430 MICROCONTROLLERS

- ➢ The MSP430 was introduced in the late 1990s
- ➢ 16bit processor with a vonNeumann architecture, designed for low-power applications
- ➢ The CPU is often described as a reduced instruction set computer (RISC)
- ➢ The registers in the CPU are also all 16 bits wide and can be used interchangeably for either data or addresses
- ➢ The MSP430 has 16registers in its CPU, which enhances efficiency because they can be used for local variables, parameters passed to subroutines, and either addresses or data



➢Several features make the MSP430 suitable for low-power and portable applications:

- The CPU is small and efficient, with a large number of registers
- It is extremely easy to put the device into a low-power mode. No special instruction is needed
- The crystal runs continuously at 32KHz and is used to wake the device periodically.
- MSP430 can wake from a stand by mode rapidly, perform its tasks, and return to a low-power mode
- A wide range of peripherals is available, many of which can run autonomously without the CPU for most of the time



- Many portable devices include liquid crystal displays which the MSP430 can drive directly.

MSP430 Family:

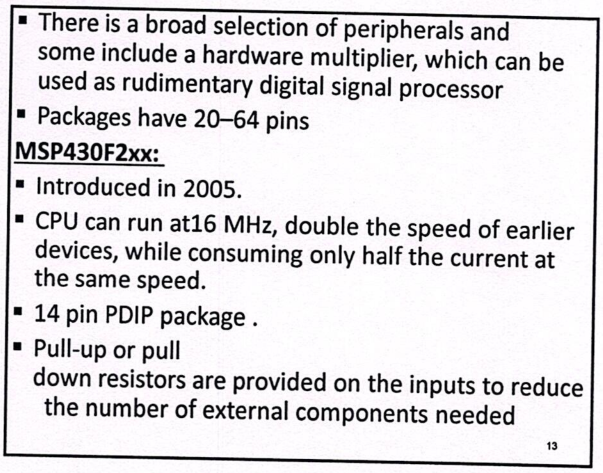
- The letter after MSP430 shows the type of memory.

F – Flash memory, C for ROM

- The second letter shows ASIP. E for Electricity, W for water, G for signals that require a gain stage.
- The next digit shows the family and the final two or three digits identify the specific device

MSP430x1xx:

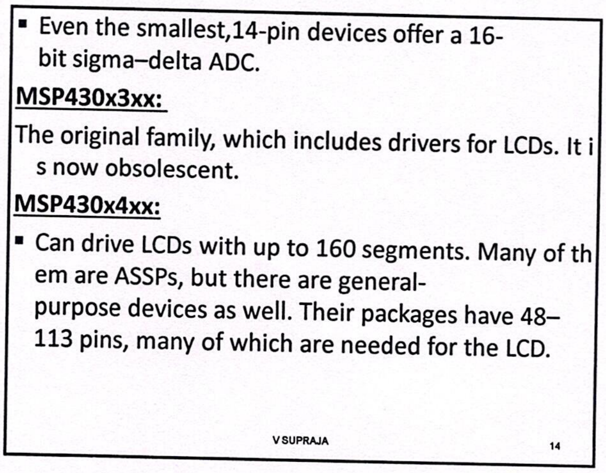
- Provides a wide range of general purpose devices from simple versions to complete systems for processing signals



- There is a broad selection of peripherals and some include a hardware multiplier, which can be used as rudimentary digital signal processor
- Packages have 20–64 pins

MSP430F2xx:

- Introduced in 2005.
- CPU can run at16 MHz, double the speed of earlier devices, while consuming only half the current at the same speed.
- 14 pin PDIP package .
- Pull-up or pull down resistors are provided on the inputs to reduce the number of external components needed



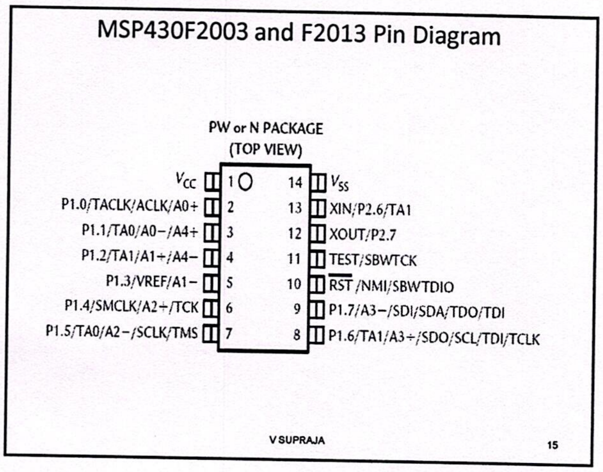
- Even the smallest,14-pin devices offer a 16-bit sigma–delta ADC.

MSP430x3xx:

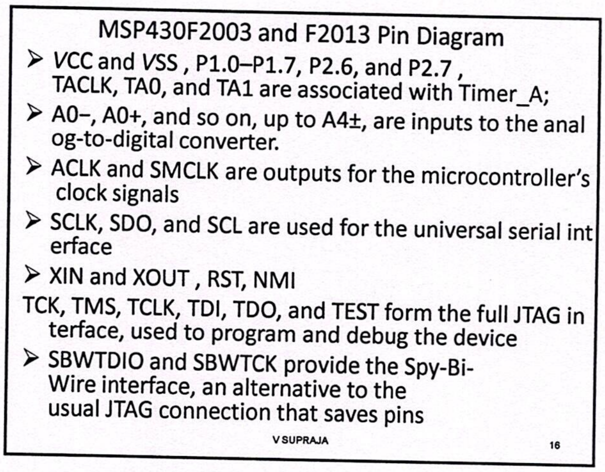
The original family, which includes drivers for LCDs. It is now obsolescent.

MSP430x4xx:

- Can drive LCDs with up to 160 segments. Many of them are ASSPs, but there are general-purpose devices as well. Their packages have 48–113 pins, many of which are needed for the LCD.



MSP430F2003 and F2013 Pin Diagram

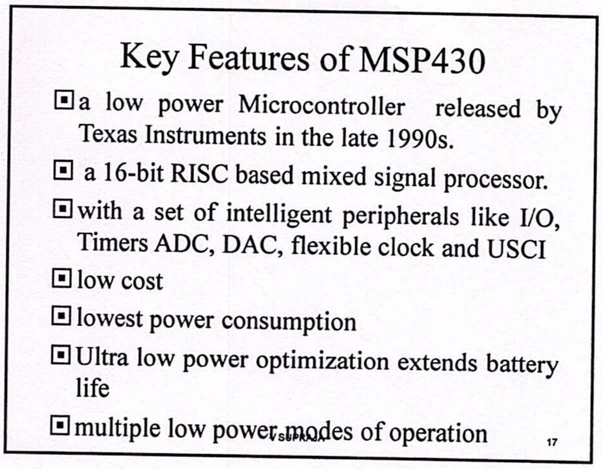


MSP430F2003 and F2013 Pin Diagram

- ➢ *VCC* and *VSS* , P1.0–P1.7, P2.6, and P2.7 , TACLK, TA0, and TA1 are associated with Timer_A;
- ➢ A0−, A0+, and so on, up to A4±, are inputs to the analog-to-digital converter.
- ➢ ACLK and SMCLK are outputs for the microcontroller's clock signals
- ➢ SCLK, SDO, and SCL are used for the universal serial interface
- ➢ XIN and XOUT , RST, NMI

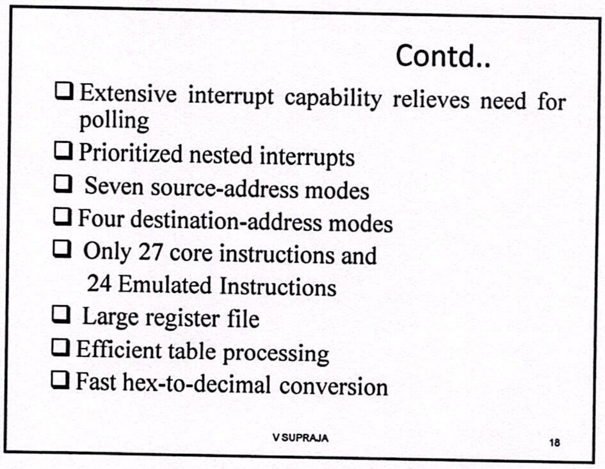
TCK, TMS, TCLK, TDI, TDO, and TEST form the full JTAG interface, used to program and debug the device

- ➢ SBWTDIO and SBWTCK provide the Spy-Bi-Wire interface, an alternative to the usual JTAG connection that saves pins



Key Features of MSP430

- ▣ a low power Microcontroller released by Texas Instruments in the late 1990s.
- ▣ a 16-bit RISC based mixed signal processor.
- ▣ with a set of intelligent peripherals like I/O, Timers ADC, DAC, flexible clock and USCI
- ▣ low cost
- ▣ lowest power consumption
- ▣ Ultra low power optimization extends battery life
- ▣ multiple low power modes of operation



Contd..

- ❑ Extensive interrupt capability relieves need for polling
- ❑ Prioritized nested interrupts
- ❑ Seven source-address modes
- ❑ Four destination-address modes
- ❑ Only 27 core instructions and 24 Emulated Instructions
- ❑ Large register file
- ❑ Efficient table processing
- ❑ Fast hex-to-decimal conversion

contd..

- MSP430 requires
  0.1 µ A for RAM data Retention,
  0.8 µ A for RTC mode operation
  250 µA /MIPS for active mode operation.
- Low operation voltage (from 1.8 V to 3.6 V).
- Zero-power Brown-Out -Reset (BOR)

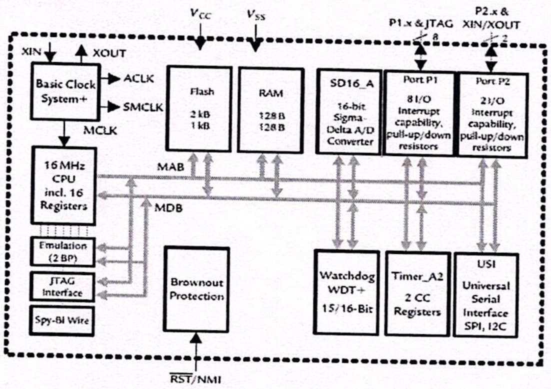
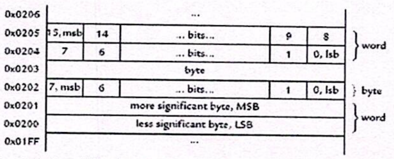


Figure 2.2: Block diagram of the MSP430F2003 and F2013, taken from the data sheet.

MEMORY



0x0206	...					
0x0205	15, msb	14	... bits...	9	8	word
0x0204	7	6	... bits...	1	0, lsb	
0x0203	byte					
0x0202	7, msb	6	... bits...	1	0, lsb	byte
0x0201	more significant byte, MSB					word
0x0200	less significant byte, LSB					
0x01FF	...					

Figure 2.3: Ordering of bits, bytes, and words in memory, adapted from the *MSP430x2xx Family User's Guide*. Addresses increase up the page.

➢The MSP430X extends the range
of memory by a factor of 16 to 20 bytes by adding a further 4 bits to the address bus and the registers in the CPU.
➢Byte accessing and Word accessing.
➢Two bytes at 0x0200 and 0x0201 can be considered as a valid word with address 0x0200

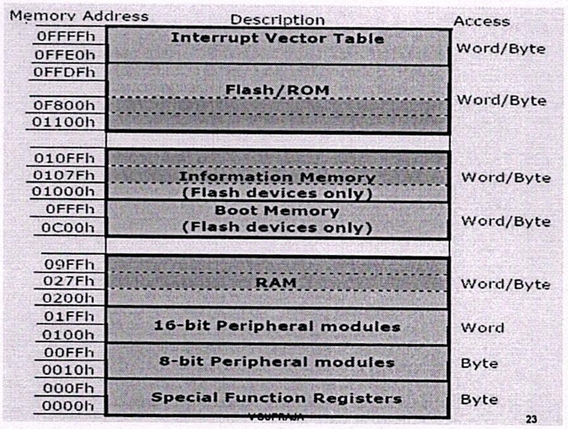
➢Hexadecimal value 0x1234

Little-endian ordering: The low-order byte is stored at the lower address and the Higherorder byte at the higher address. This is used byte MSP430 and is the more common format.

Big-endian ordering: The high-order byte is stored at the lower address. This is use dby the Freescale HCS08.

➢Addresses increase from left to right across each line. This means that the low-order byte is displayed first, followed by the high-order byte. Thus our value of

0x1234 is displayed as 34 12.



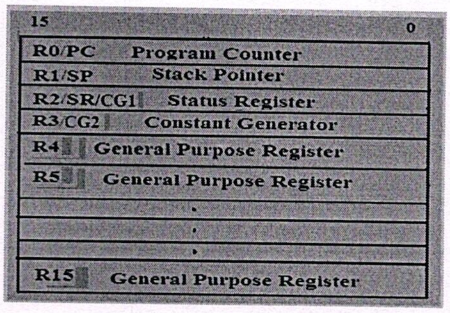
Memory Address	Description	Access
0FFFFh 0FFE0h	Interrupt Vector Table	Word/Byte
0FFDFh 0F800h 01100h	Flash/ROM	Word/Byte
010FFh 0107Fh 01000h	Information Memory (Flash devices only)	Word/Byte
0FFFh 0C00h	Boot Memory (Flash devices only)	Word/Byte
09FFh 027Fh 0200h	RAM	Word/Byte
01FFh 0100h	16-bit Peripheral modules	Word
00FFh 0010h	8-bit Peripheral modules	Byte
000Fh 0000h	Special Function Registers	Byte

MSP430 CPU

- ▣ The CPU of MSP 430 includes a 16-bit ALU and a set of 16 Registers R0 –R15.In these registers Four are special Purpose and 12 are general purpose registers . All the registers can be addressed in the same way.

  The special Purpose Registers are
- ▣ PC (Program Counter), SP (Stack Pointer) , SR (Status Register) and CGx (Constant Generator)

Registers in the CPU of the MSP430


15
0
R0/PC Program Counter
R1/SP Stack Pointer
R2/SR/CG1 Status Register
R3/CG2 Constant Generator
R4 General Purpose Register
R5 General Purpose Register
R15 General Purpose Register

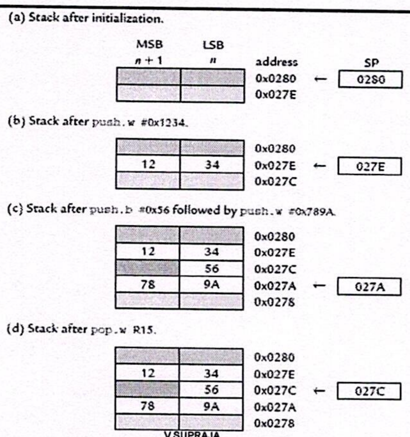
- ▣ The MSP430 CPU includes an arithmetic logic unit (ALU) that handles addition, subtraction, comparison and logical (AND, XOR) operations. ALU operations can affect the overflow, zero, negative, and carry flags in the status register.

R0: Program Counter (PC)

- ▣ The 16-bit Program Counter (PC/R0) points to the next instruction to be read from memory and executed by the CPU. The Program counter is incremented by 2. It is important to note that the PC is aligned at even addresses, because the instructions are 16 bits, even though the individual memory addresses contain 8-bit values.
- ▣ Subroutines and interrupts also modify the PC but in these cases the previous value is saved on the stack and restored later

R1: Stack Pointer (SP)

- The Stack Pointer (SP/R1) is located in R1.
- Stack can be used by user to store data for later use(instructions: store by PUSH, retrieve by POP)
- Stack is also heavily used for temporary variables, passing parameters to subroutines and returning the result.
- The stack is allocated at top of RAM and grows down towards the low address. SP holds the address of top of the stack.


(a) Stack after initialization.

(b) Stack after push.w #0x1234.

(c) Stack after push.b #0x56 followed by push.w #0x789A.

(d) Stack after pop.w R15.

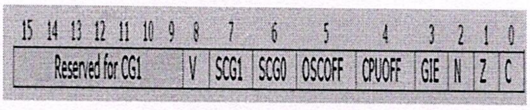
Figure 5.2: Operation of the stack in the MSP430F2013, whose RAM lies from

Contd..

- For programs written in C, the compiler initializes the stack automatically as part of the startup code, which runs silently before the program starts, but *you must initialize SP yourself in assembly language.*

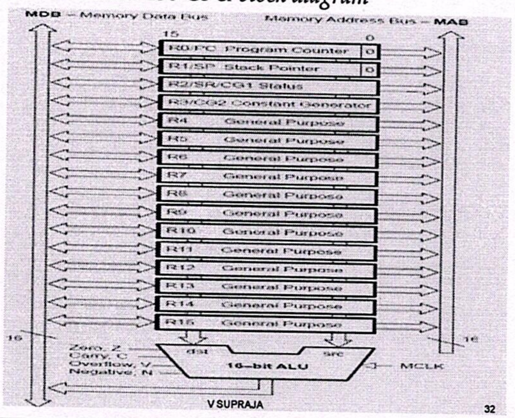
R2: Status Register (SR)

The Status Register (SR/R2) is a 16 bit register , and it stores the state and control bits. The system flags are changed automatically by the CPU depending on the result of an operation in a register. The reserved bits of the SR are used to support the constants generator.



- The hexadecimal sum 0x75+0xC7=0x13C, where the result is too large to be held in a single byte
- The *zero flag Z is set when the result of an operation is 0.*
- The *negative flag N is made equal to the msb of the result, which indicates a* negative number if the values are signed.
- The *signed over low flag V is set when the result of a signed operation has* overflowed, even though a carry may not be generated
- Remember that a byte can hold the values 0 to 0xFF if it is unsigned or −0x80 to 0x7F if it is signed.

MSP430 CPU block diagram



Enable Interrupts

- Setting the *general interrupt enable or GIE bit enables maskable interrupts, provided that* the individual sources of interrupts have themselves been enabled.
- Clearing the bit disables all maskable interrupts.

Control of Low-Power Modes

- The CPUOFF, OSCOFF, SCG0, and SCG1 bits control the mode of operation of the MCU. All systems are fully operational when all bits are clear. Setting combinations of these bits puts the device into one of its low-power modes.

R2/R3: Constant Generator Registers (CG1/CG2)

- Depending on the source-register addressing modes (As) value, six commonly used constants can be generated without a code word or code memory access to retrieve them. This is a very powerful feature, which allows the implementation of emulated instructions, for example, instead of implementing a core instruction for an increment, the constant generator is used.

R4 - R15: General–Purpose Registers

- These general-purpose registers are used to store data values , address pointers, or index values and can be accessed with byte or word instructions.

Addressing modes

The MSP430 supports seven addressing modes for the source operand and four addressing modes for the destination operand . They are

- Register mode
- Indexed mode
- Symbolic mode
- Absolute mode
- Indirect register mode
- Indirect auto incrément mode
- Immediate mode

Register Mode

- Register mode operations work directly on the processor registers, R4 through R15, or on special function registers, such as the program counter or status register. They are very efficient in terms of both instruction speed and code space.

```
Ex :        MOV.b R4, R5
            MOV.W R4,R5
```

Move (copy) the contents of source (register R4) to destination (register R5). Register R4 is not affected.

Indexed mode

- The Indexed mode commands are formatted as X(Rn), where X is a constant and Rn is one of the CPU registers. The absolute memory location X+Rn is addressed.
- Indexed mode addressing is useful for applications such as lookup tables

  Ex : MOV. b F000h(R5), R4

  Move (copy) the contents at source address (F000h +R5) to destination (register R4)

Symbolic mode

- ▣ Symbolic mode allows the assignment of labels to fixed memory locations, so that those locations can be addressed. This is useful for the development of embedded programs.
- ▣ MOV XPT, YPT; Move the content of source address XP (x pointer) to the destination address YPT (y pointer).
- ▣ MOV.w LoopCtr, R6; Load word loopCtr into R6
- ▣ Assemler replaces this by the indexed form
- ▣ Mov.w X(PC), R6; Load word loopCtr into R6
- ▣ Where X= LoopCtr-PC

Absolute mode

- Similar to Symbolic mode, with the difference that the label is preceded by “&”.

  The word following the instruction contains the absolute address. X is stored in the next word. Indexed mode X(SR) is used

  MOV &XPT, &YPT; Move the content of source address XPT to the destination address YPT.

Eg1: mov.b &P1IN, R6; Load byte P1IN into R6.

Assemler replaces this by the indexed form

Mov.b P1IN(SR), R6; Load byte P1IN into R6.

Assemler replaces this by the indexed form.

P1IN is the absolute address of the register.

SP- Relative Mode

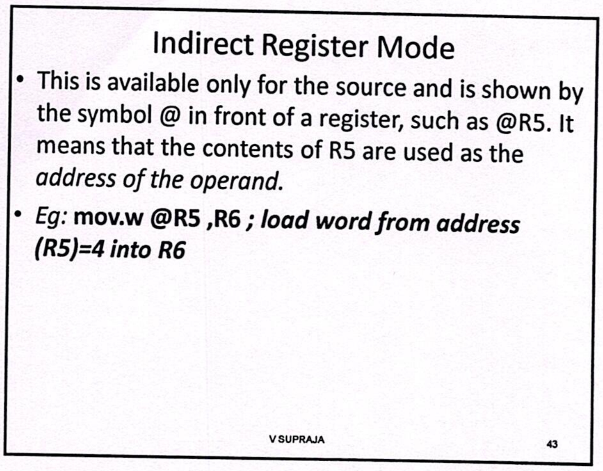
- The stack pointer SP can be used as the register in indexed mode like any other
- Suppose that we wanted to copy the value that had been pushed onto the stack before the most recent one

Eg: mov.w 2(SP),R6 ; *copy most recent word but one fr*

- For example, suppose that the stack were as shown in Figure 5.2(d) with SP=0x027C. Then the preceding instruction would load 0x1234 into R6.

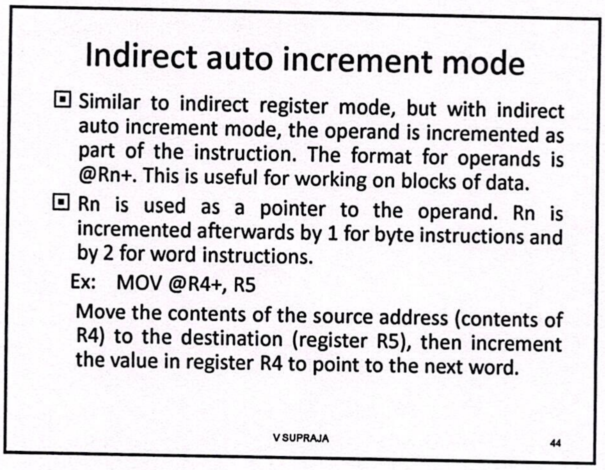
Indirect register mode

- The data word addressed is located in the memory location pointed to by Rn. Indirect mode is not valid for destination operands, but can be emulated with the indexed mode format @(Rn). Here Rn is used as a pointer to the operand.
- MOV @(R4), R5
- Move the contents of the source address (contents of R4) to the destination (register R5). Register R4 is not modified



Indirect Register Mode

- This is available only for the source and is shown by the symbol @ in front of a register, such as @R5. It means that the contents of R5 are used as the *address of the operand.*
- *Eg:* **mov.w @R5 ,R6 ;** ***load word from address (R5)=4 into R6***

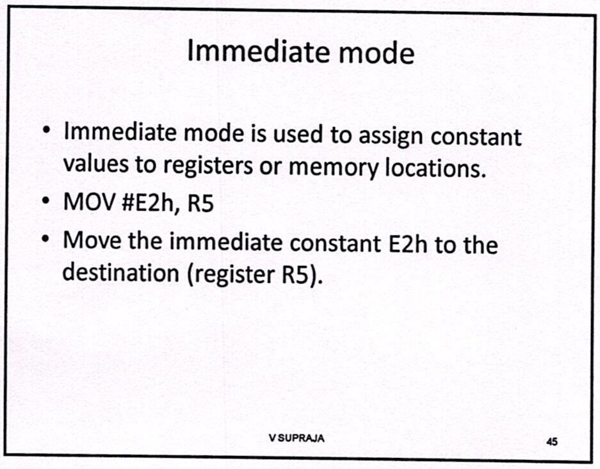


Indirect auto increment mode

- ▣ Similar to indirect register mode, but with indirect auto increment mode, the operand is incremented as part of the instruction. The format for operands is @Rn+. This is useful for working on blocks of data.
- ▣ Rn is used as a pointer to the operand. Rn is incremented afterwards by 1 for byte instructions and by 2 for word instructions.

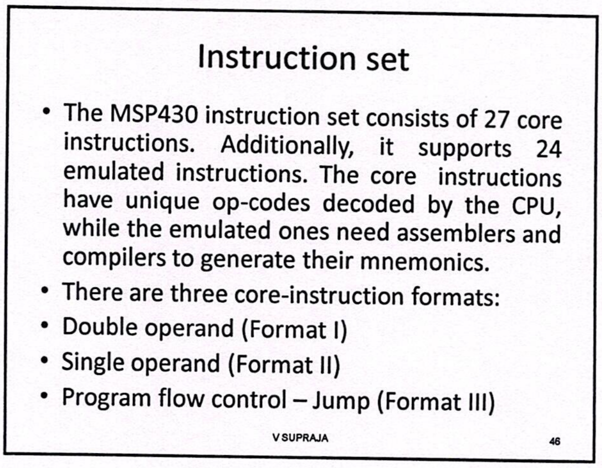
Ex: MOV @R4+, R5

Move the contents of the source address (contents of R4) to the destination (register R5), then increment the value in register R4 to point to the next word.



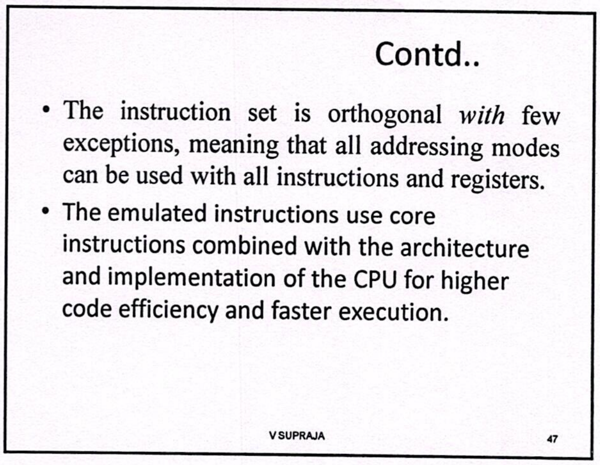
Immediate mode

- Immediate mode is used to assign constant values to registers or memory locations.
- MOV #E2h, R5
- Move the immediate constant E2h to the destination (register R5).



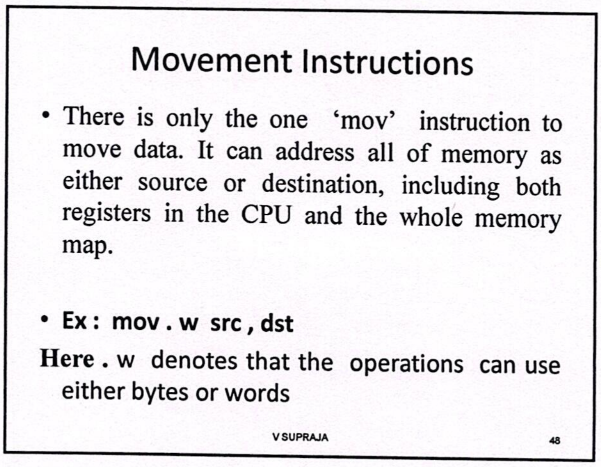
Instruction set

- The MSP430 instruction set consists of 27 core instructions. Additionally, it supports 24 emulated instructions. The core instructions have unique op-codes decoded by the CPU, while the emulated ones need assemblers and compilers to generate their mnemonics.
- There are three core-instruction formats:
- Double operand (Format I)
- Single operand (Format II)
- Program flow control – Jump (Format III)



Contd..

- The instruction set is orthogonal *with* few exceptions, meaning that all addressing modes can be used with all instructions and registers.
- The emulated instructions use core instructions combined with the architecture and implementation of the CPU for higher code efficiency and faster execution.



Movement Instructions

- There is only the one ‘mov’ instruction to move data. It can address all of memory as either source or destination, including both registers in the CPU and the whole memory map.
- **Ex : mov . w src , dst**

Here . w denotes that the operations can use either bytes or words

Stack Operations

- These instructions either push data onto the stack or pop them off .

ex 1: push .w src ; push data onto stack

ex 2 : pop .w dst ; pop data off stack.

The pop operation is emulated using post-increment addressing but push requires a special instruction because pre-decrement addressing is not available.

Arithmetic and Logic Instructions with Two Operands

▣ add.w src ,dst ; add
▣ addc.w src ,dst *; add with carry*
▣ adc.w dst *; add carry bit*
▣ sub.w src ,dst *; subtract*
▣ subc.w src ,dst *; subtract with borrow*
▣ sbc.w dst *; subtract borrow bit*
▣ cmp.w src ,dst *; compare , set flags only.*

The compare operation cmp is the same as subtraction except that only the bits in SR are affected ; the result is not written back to the destination.

Arithmetic Instructions with One Operand

- All these are instructions are emulated, which means that the operand is always a destination.
- Ex: clr.w dst *; clear*
- dec.w dst *; decrement*
- decd.w dst *; double decrement*
- inc.w dst *; increment*
- incd.w dst *; double increment*
- tst.w dst *; test (compare with 0)*

Decimal Arithmetic

- These instructions are used when operands are binary-coded decimal (BCD) rather than ordinary binary values.
- Ex : dadd.w src , dst ; decimal add with carry.
- dadc.w dst ; decimal add carry bit

Logic Instructions with Two Operands

- The MSP430 has the usual and and exclusive-OR xor instructions. The and and bitwise test operations are identical except that bit is only a test and does not change its destination.
- Ex : and.w src ,dst *; bitwise and .*
- xor.w src ,dst *; bitwise exclusive or*
- bit.w src ,dst *; bitwise test , set flags only*
- bis.w src ,dst *; bit set; Not.src.and.dst*
- bic.w src ,dst *; bit clear;src.or.dst*

Logic Instructions with One Operand

- There is only one instruction of this type . invert ‘inv’ instruction , also known as ones complement ,which changes all 0 bits to 1 and 1s to 0.
- Ex : inv.w dst ; invert bits

Byte Manipulation

- ▣ These instructions do not need a suffix because the size of. the operands is fixed.
- ▣ Ex : Swpb src ; swap upper and lower bytes (word only)
- ▣ Ex : sxt src *; extend sign of lower byte (word only)*
- ▣ The swap bytes instruction 'swpb' swaps the two bytes in a word.
- ▣ The sign extend instruction sxt is used to convert a signed byte into a signed word.

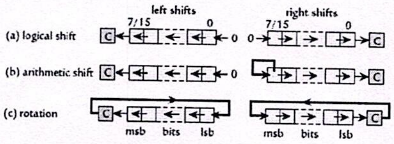
Operations on Bits in Status Register

- ▣ There is a set of emulated instructions to set or clear the four lowest bits in the status register and these can be masked using the constant generator.
- ▣ Ex: clrc ; clear carry bit.
- ▣ clrn ; clear negative bit.
- ▣ clrz ; clear zero bit.
- ▣ setc ; set carry bit.
- ▣ setn ; set negative bit.
- ▣ **setz ; set zero bit.**
- ▣ **dint ; disable general interrupts.**
- ▣ **eint ; enable general interrupts.**

Shift and Rotate Instructions

There are three types of shifts

(i) logical shift (ii) arithmetic shift (iii) rotation. They are explained below.



Contd..

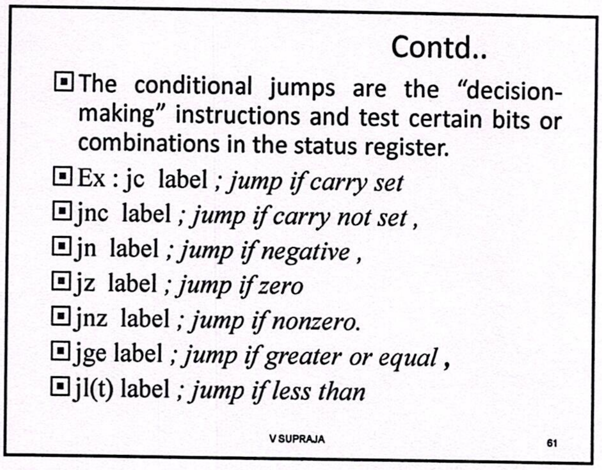
- ▣ Logical shift inserts zeroes for both right and left shifts.
- ▣ Arithmetic shift inserts zeroes for left shifts but the most significant bit, which carries the sign, is replicated for right shifts.
- ▣ Rotation does not introduce or lose any bits; bits that are moved out of one end of the register are passed around to the other.
- ▣ Ex : rla dst *; arithmetic shift left*
- ▣ rra src *; arithmetic shift right.*
- ▣ rlc dst ; rotate left through carry.
- ▣ rrc src *; rotate right through carry*

Flow of Control

- The most common elementary use of call is for a subroutine that begins at a particular label.
- Ex: br src *; branch (go to).*
- call src *; call subroutine.*
- ret *; return from subroutine.*
- reti *; return from interrupt.*
- nop *; no operation (consumes single cycle)*

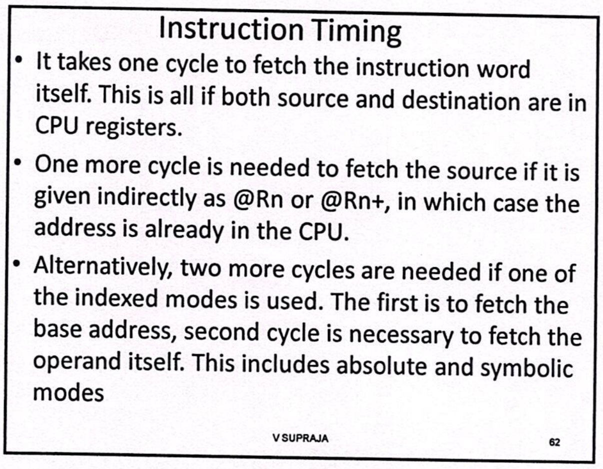
Jumps(Unconditional and Conditional)

- jmp fits in a single word, including the offset, but its range is limited to about ±1KB from the current location.
- jmp label *; unconditional jump.*



Contd..

- The conditional jumps are the "decision-making" instructions and test certain bits or combinations in the status register.
- Ex : jc label *; jump if carry set*
- jnc label *; jump if carry not set ,*
- jn label *; jump if negative ,*
- jz label *; jump if zero*
- jnz label *; jump if nonzero.*
- jge label *; jump if greater or equal ,*
- jl(t) label *; jump if less than*



Instruction Timing

- It takes one cycle to fetch the instruction word itself. This is all if both source and destination are in CPU registers.
- One more cycle is needed to fetch the source if it is given indirectly as @Rn or @Rn+, in which case the address is already in the CPU.
- Alternatively, two more cycles are needed if one of the indexed modes is used. The first is to fetch the base address, second cycle is necessary to fetch the operand itself. This includes absolute and symbolic modes

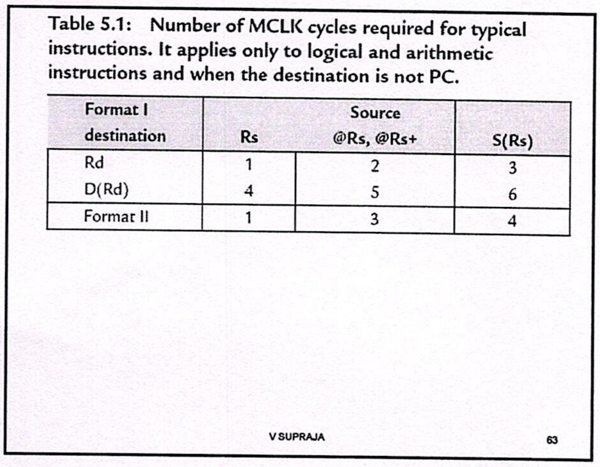
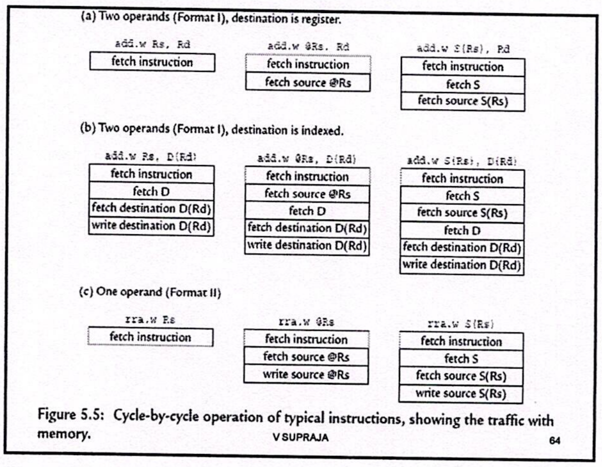


Table 5.1: Number of MCLK cycles required for typical instructions. It applies only to logical and arithmetic instructions and when the destination is not PC.

Format I destination	Rs	Source @Rs, @Rs+	S(Rs)
Rd	1	2	3
D(Rd)	4	5	6
Format II	1	3	4



(a) Two operands (Format I), destination is register.

add.w Rs, Rd	add.w @Rs, Rd	add.w S(Rs), Rd
fetch instruction	fetch instruction	fetch instruction
	fetch source @Rs	fetch S
		fetch source S(Rs)

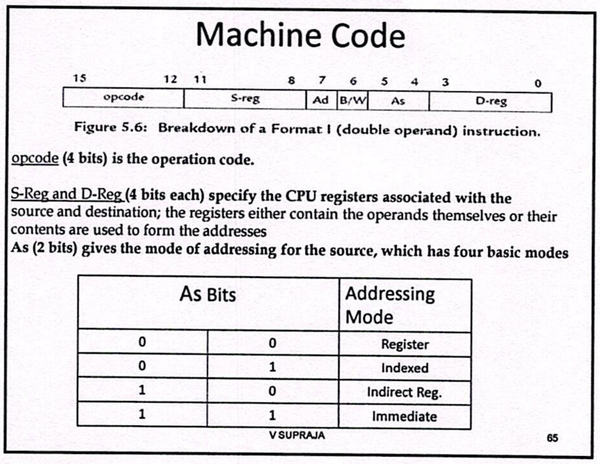
(b) Two operands (Format I), destination is indexed.

add.w Rs, D(Rd)	add.w @Rs, D(Rd)	add.w S(Rs), D(Rd)
fetch instruction	fetch instruction	fetch instruction
fetch D	fetch source @Rs	fetch S
fetch destination D(Rd)	fetch D	fetch source S(Rs)
write destination D(Rd)	fetch destination D(Rd)	fetch D
	write destination D(Rd)	fetch destination D(Rd)
		write destination D(Rd)

(c) One operand (Format II)

rra.w Rs	rra.w @Rs	rra.w S(Rs)
fetch instruction	fetch instruction	fetch instruction
	fetch source @Rs	fetch S
	write source @Rs	fetch source S(Rs)
		write source S(Rs)

Figure 5.5: Cycle-by-cycle operation of typical instructions, showing the traffic with memory.



Machine Code

15–12	11–8	7	6	5–4	3–0
opcode	S-reg	Ad	B/W	As	D-reg

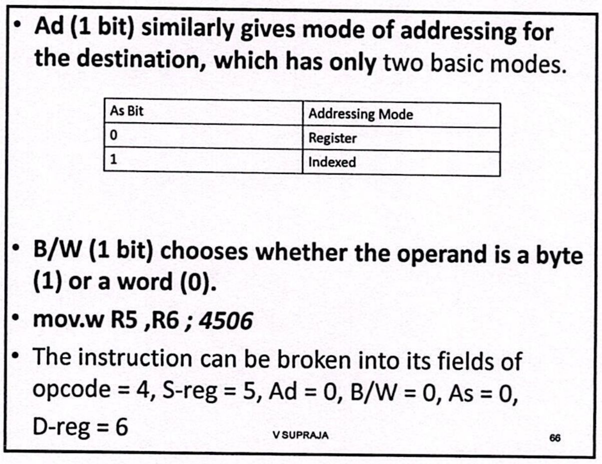
Figure 5.6: Breakdown of a Format I (double operand) instruction.

opcode (4 bits) is the operation code.

S-Reg and D-Reg (4 bits each) specify the CPU registers associated with the source and destination; the registers either contain the operands themselves or their contents are used to form the addresses

As (2 bits) gives the mode of addressing for the source, which has four basic modes

As Bits		Addressing Mode
0	0	Register
0	1	Indexed
1	0	Indirect Reg.
1	1	Immediate



- **Ad (1 bit) similarly gives mode of addressing for the destination, which has only** two basic modes.

As Bit	Addressing Mode
0	Register
1	Indexed

- **B/W (1 bit) chooses whether the operand is a byte (1) or a word (0).**
- **mov.w R5 ,R6 ; *4506***
- The instruction can be broken into its fields of opcode = 4, S-reg = 5, Ad = 0, B/W = 0, As = 0, D-reg = 6

- The opcode of 4 represents a move.
- The bit B/W = 0 shows that the operand is a word.
- The addressing mode for the source is As = 0, which is register. The register is S-reg = 5, which is R5 as expected.
- Similarly, the addressing mode for the destination is Ad = 0, which again means register. The register is D-reg = 6 = R6.

Eg2: **add.w R5 ,R6 ; *5506***

Eg3: **mov.w #5,R6 ; *4036 0005***

▣ Now there are two words. The fields of the instruction are opcode = 4, S-reg = 0, Ad = 0, B/W = 0, As = 3 = 11b, D-reg = 6. The difference is in the specification of the source, which means auto increment. The register is Sreg = 0, which is the PC

Eg3: mov.b #4,R6 ; *4266*

- This breaks into opcode = 4, S-reg = 2, Ad = 0, B/W = 1, As = 2 = 10b, D-reg = 6. The B/W bit flags a byte rather than a word. The source appears to have indirect register mode on R2/SR/CG1, but this is translated by the constant generator into a value of 0x0004, as required.

ILLEGAL INSTRUCTIONS

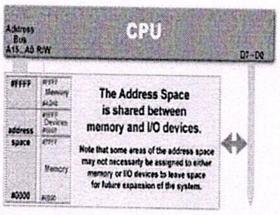
▣ a reset is generated if the CPU tries to fetch an instruction from the range of memory allocated to the special function and peripheral registers (addresses below 0x0200)

▣ Data can be fetched from a nonexistent address: empty regions of the memory map. Execution proceeds as normal but the "fetched" value is random.

▣ Data for a word should be aligned to even addresses but you could attempt to fetch a word from an odd address

▣ Accessing peripheral registers with word access. It is possible to read the lower byte alone but not the upper Byte. An attempt to read the upper byte returns the lower byte instead.

Memory Mapped Input and Output


CPU
Address Bus A15...A0 R/W
D7...D0
#FFFF
Memory
Devices
address space
Memory
#0000
The Address Space is shared between memory and I/O devices.
Note that some areas of the address space may not necessarily be assigned to either memory or I/O devices to leave space for future expansion of the system.

➢The MSP430 uses *memory-mapped input* and output. This means that the ports simply appear to the CPU as particular memory registers called *peripheral registers*

➢These registers can be read, written and modified in almost the same way as simple registers in RAM. You can even do arithmetic with them, with some restrictions

- **Port P1 input, P1IN:** Reading returns the logical values on the inputs if they are configured for digital input and output. This register is read-only. It is also *volatile,* which means that it may change at a time that a program cannot predict.
- **Port P1 output, P1OUT:**Writing sends the value to be driven onto the pin if it is configured as a digital output. If the pin is not currently an output, the value is stored in a buffer and appears on the pin if it is later switched to be an output.
- **Port P1 direction, P1DIR: A bit of 0 configures the pin as an input, which is the** default. Writing a 1 switches the pin to become an output.

MSP430F5529IPNFunctional Block Diagram

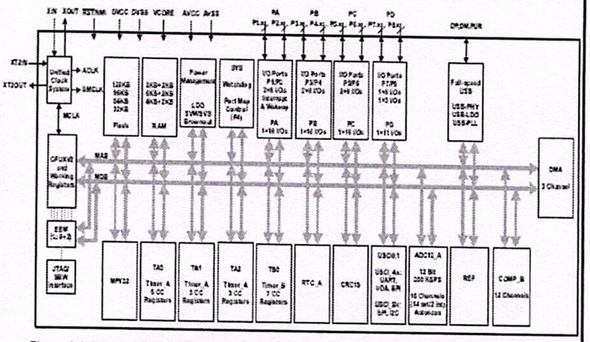
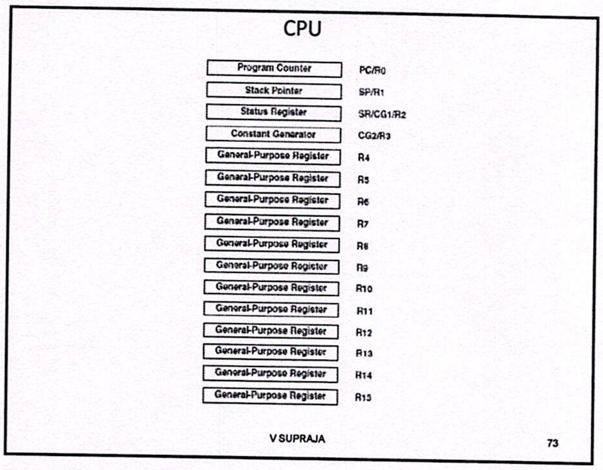

XIN XOUT RST/NMI DVCC DVSS VCORE AVCC AVSS
PA PB PC PD
P1.x, P2.x, P3.x, P4.x, P5.x, P6.x, P7.x, P8.x
DP,DM,PUR
XT2IN
XT2OUT
Unified Clock System
ACLK
SMCLK
MCLK
128KB 96KB 64KB 32KB Flash
8KB+2KB 6KB+2KB 4KB+2KB RAM
Power Management LDO SVM/SVS Brownout
SYS Watchdog Port Map Control (P4)
I/O Ports P1/P2 2×8 I/Os Interrupt & Wakeup PA 1×16 I/Os
I/O Ports P3/P4 2×8 I/Os PB 1×16 I/Os
I/O Ports P5/P6 2×8 I/Os PC 1×16 I/Os
I/O Ports P7/P8 1×8 I/Os 1×3 I/Os PD 1×11 I/Os
Full-speed USB USB-PHY USB-LDO USB-PLL
CPUXV2 and Working Registers
MAB
MDB
DMA 3 Channel
EEM (L: 8+2)
JTAG/ SBW Interface
MPY32
TA0 Timer_A 5 CC Registers
TA1 Timer_A 3 CC Registers
TA2 Timer_A 3 CC Registers
TB0 Timer_B 7 CC Registers
RTC_A
CRC16
USCI0,1 USCI_Ax: UART, IrDA, SPI USCI_Bx: SPI, I2C
ADC12_A 12 Bit 200 KSPS 16 Channels (14 ext/2 int) Autoscan
REF
COMP_B 12 Channels

Figure 1-1. Functional Block Diagram – MSP430F5529IPN, MSP430F5527IPN, MSP430F5525IPN, MSP430F5521IPN



CPU

Program Counter	PC/R0
Stack Pointer	SP/R1
Status Register	SR/CG1/R2
Constant Generator	CG2/R3
General-Purpose Register	R4
General-Purpose Register	R5
General-Purpose Register	R6
General-Purpose Register	R7
General-Purpose Register	R8
General-Purpose Register	R9
General-Purpose Register	R10
General-Purpose Register	R11
General-Purpose Register	R12
General-Purpose Register	R13
General-Purpose Register	R14
General-Purpose Register	R15