ХНУРЕ

Системи супутникового зв’язку та навігації

Конспект лекцій

Доцент МЗ Кривенко С.А.

2011

Харків

ЗМІСТ

Вступ

Навчальна дисципліна «Системи супутникового зв’язку та навігації» читається в восьмому семестрі студентам четвертого курсу, які готуються за напрямом 6.050903 «Телекомунікації».

Предметом дисципліни є побудова та функціонування систем супутникового зв’язку та навігації.

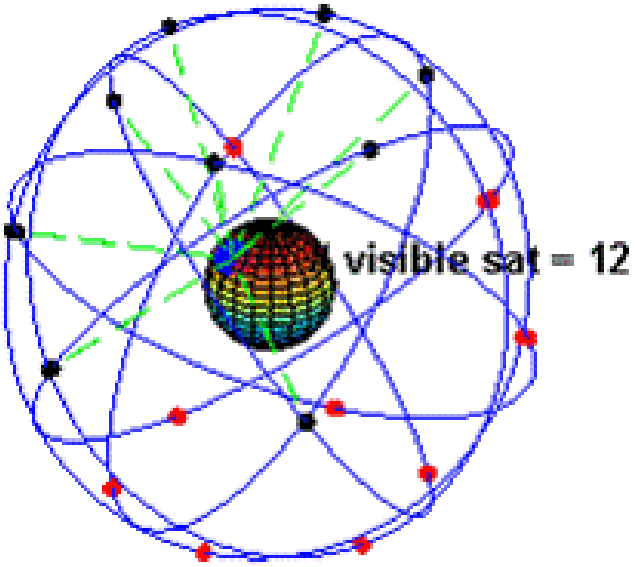
Мета дисципліни полягає у формуванні знань і умінь розробки і експлуатації апаратних та програмних засобів систем супутникового зв’язку та навігації на рівні, достатньому для практичної діяльності за спеціальністю.

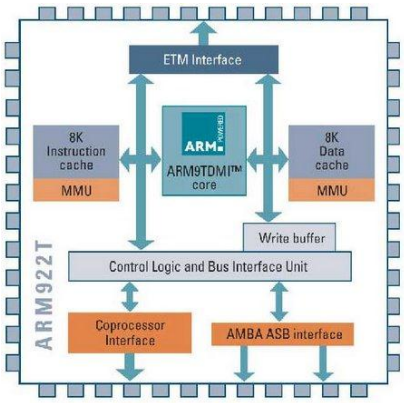
Тематичний модуль 1.
Принцип дії систем

Лекція 1. Ввідна. Загальні відомості

Лекція 2. Модель мікропроцесора ARM

Лекція 3. Принцип дії GPS



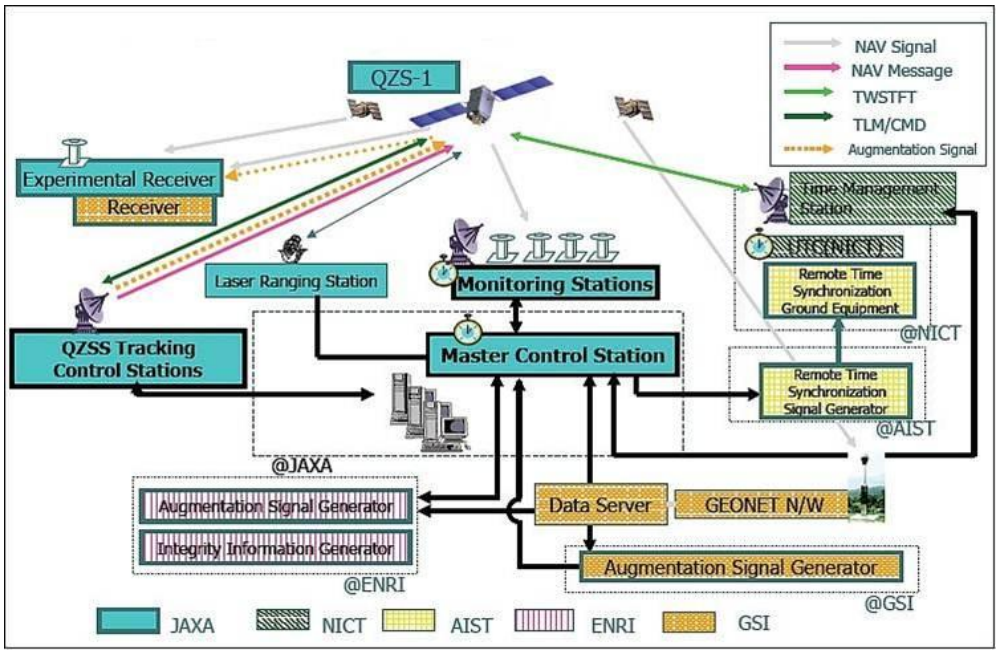


1 ВВІДНА. ЗАГАЛЬНІ ВІДОМОСТІ

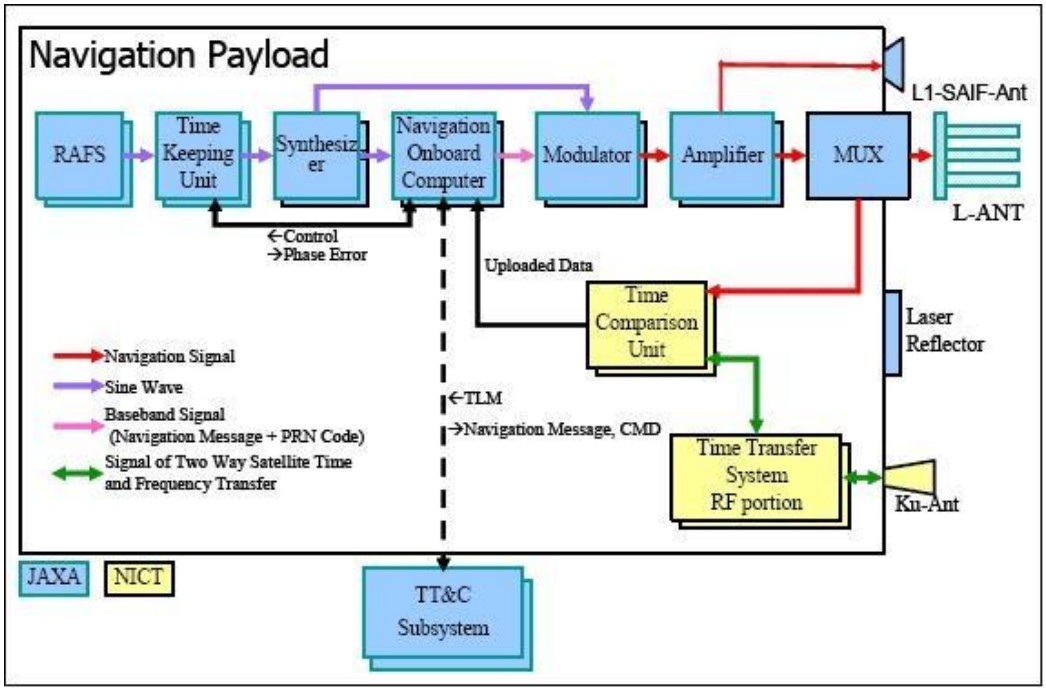
Agenda

- Принцип дії
- Реалізація систем

1.1 Принцип дії



1.2 Реалізація систем



Системи супутникового зв’язку можуть бути реалізовані на мікропроцесорі ARM, який вбудований в мікросхему FPGA(http://www.arm.com/fpga/index.html)

FPGA Device Compatibility	Implementation Tool Compatibility
Actel ProASIC3L & ProASIC3/E	Actel Libero
Actel Fusion	
Actel IGLOO/e	
Altera Cyclone II	Altera Quartus II
Altera Cyclone III	
Altera Stratix II	Synplicity Synplify Pro
Altera Stratix III	
Xilinx Spartan-3	Mentor Precision
Xilinx Virtex-2	
Xilinx Virtex-4	Xilinx ISE
Xilinx Virtex-5	

Загальні відомості щодо ARM.

ARM є сімейство мікропроцесорів архітектури RISC, які поділяють ті ж

принципи дизайну і загального набору інструкцій.

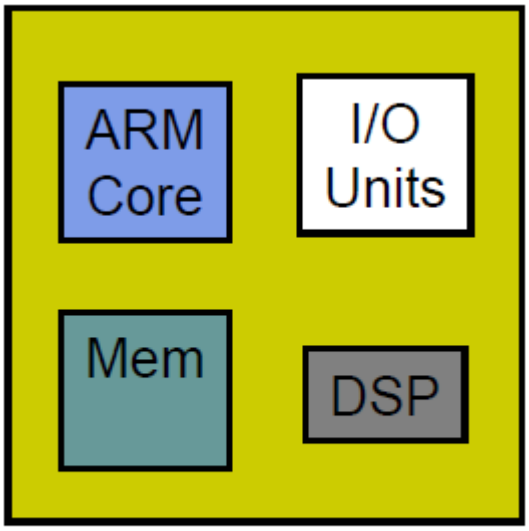
ARM сам не виробляє процесори, але надає ліцензії, іншим виробникам інтегрувати їх у свою власну систему.

Ядро ARM (CORE) це частина системи-на-кристалі.

Крім GPS, ядро ARM широко використовується в мобільних телефонах, портативних помічниках, і багатьох інших портативних пристроях побутової електроніки.

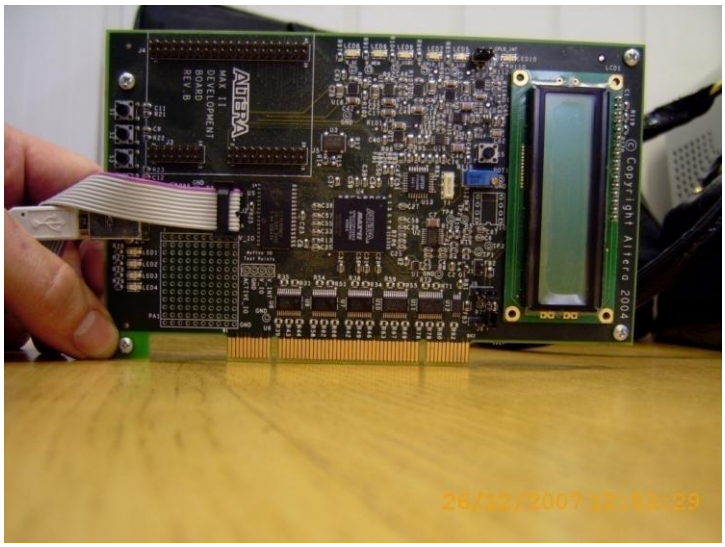
Залежно від застосування процесорів ARM доступні, наприклад: різні розміри кеша; шини різної ширини; різні тактові частоти.

Різні версії використовують різні архітектури, наприклад, ARM 7: VON NEUMANN; ARM 9: HARVARD. Мова програмування не залежать від базової архітектури.



ASIC

Приклад комплекту для проектування.



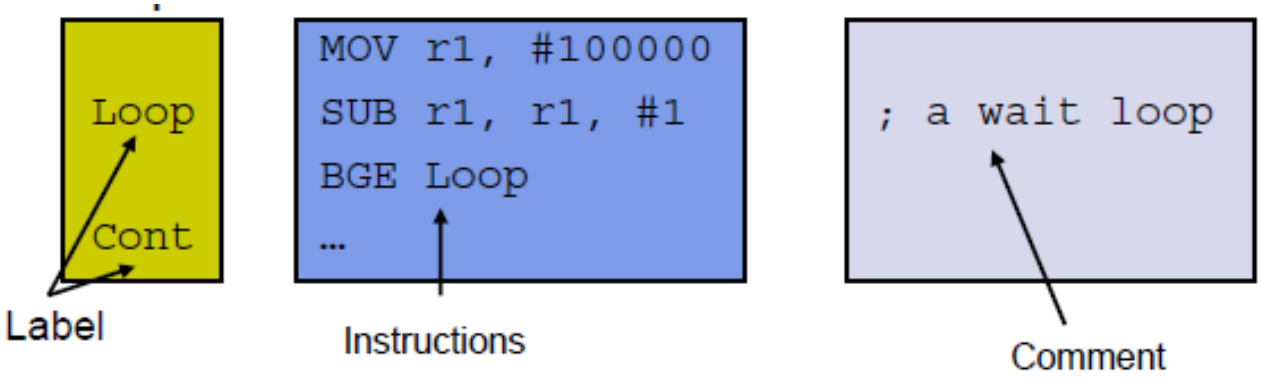
2 МОДЕЛЬ МІКРОПРОЦЕСОРА ARM

Agenda:

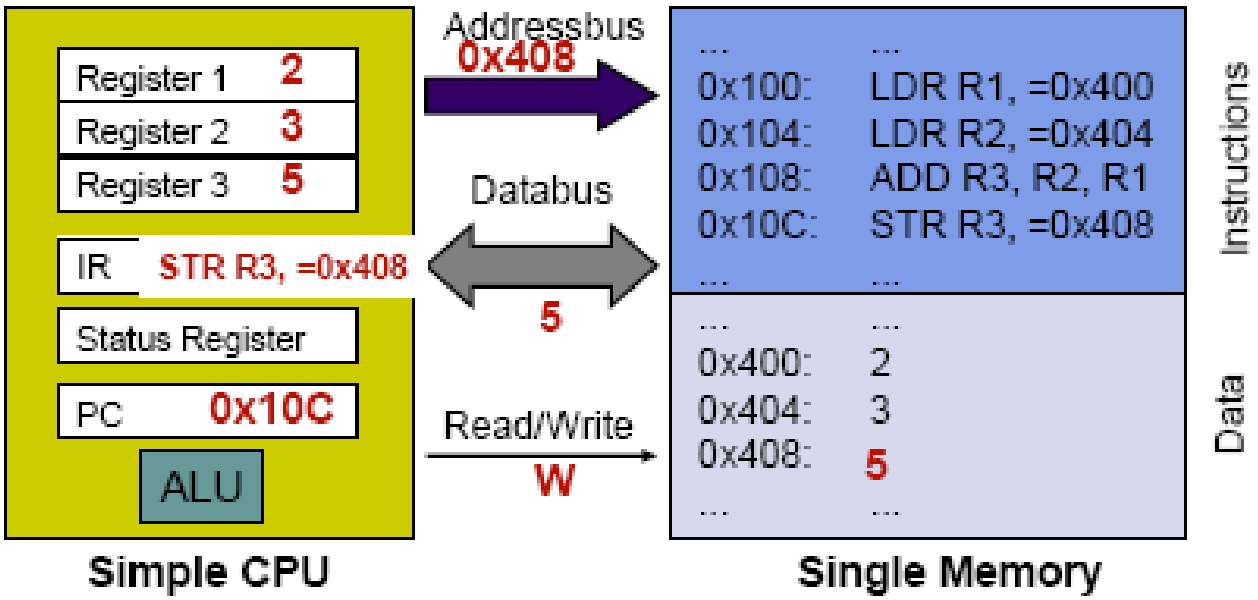
- 1) архітектура ARM;
- 2) набір інструкцій.

2.1 Архітектура ARM

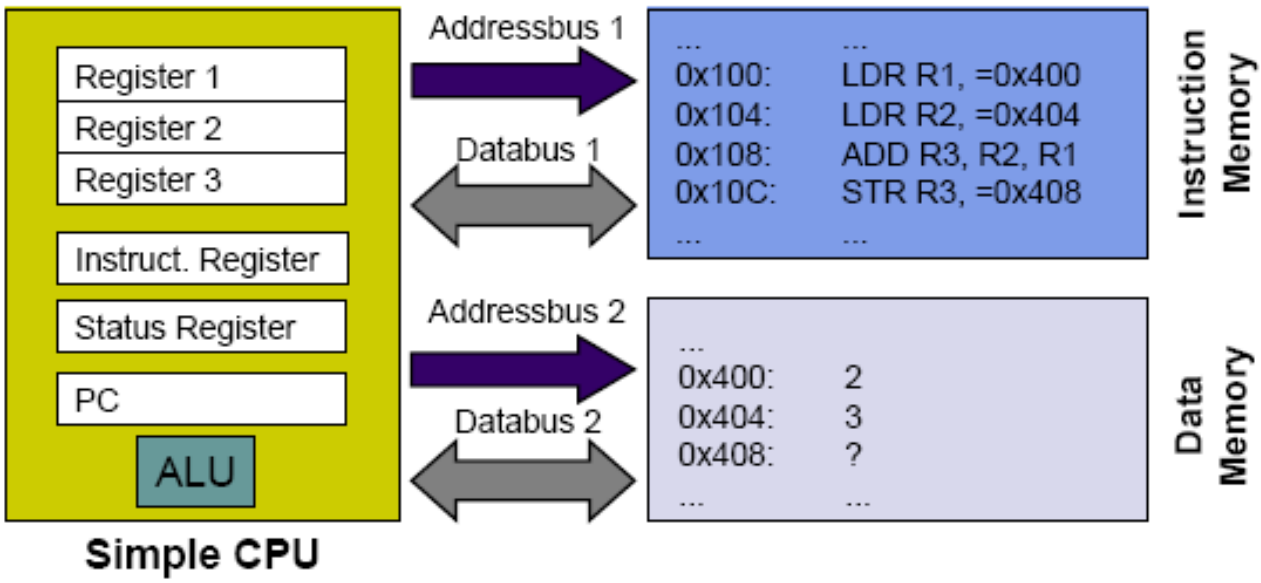
ARM асемблер



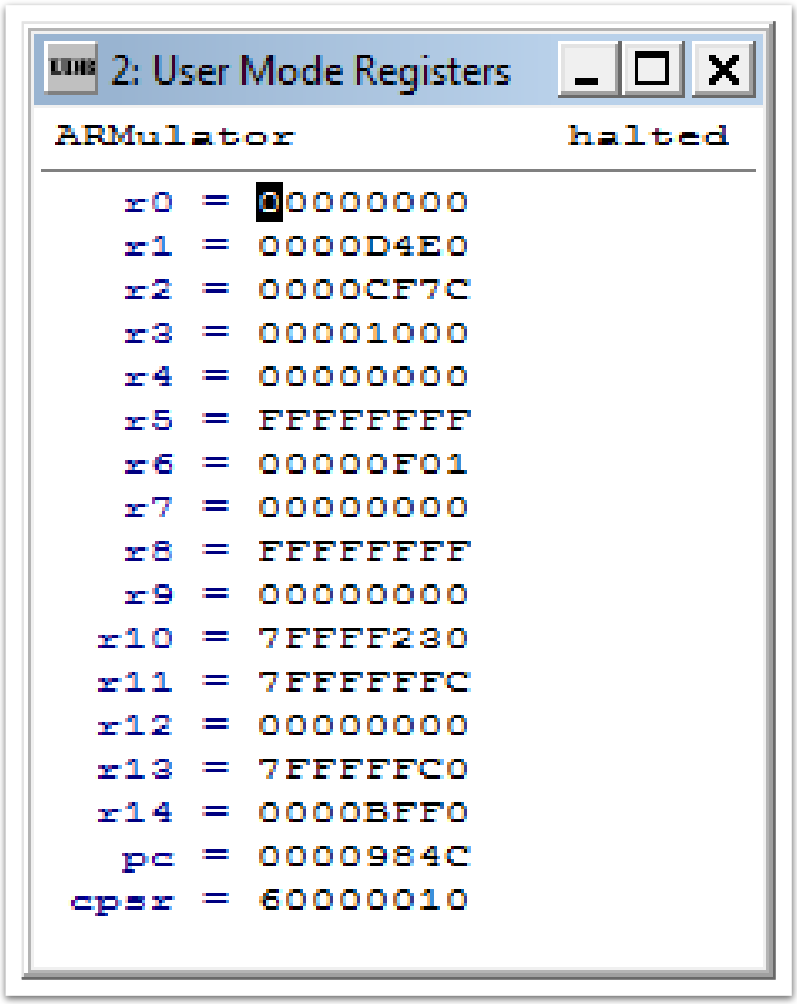
Архітектура Von Neumann складається з центрального процесора і єдиної пам'яті, пам'ять зберігає інструкції і дані.

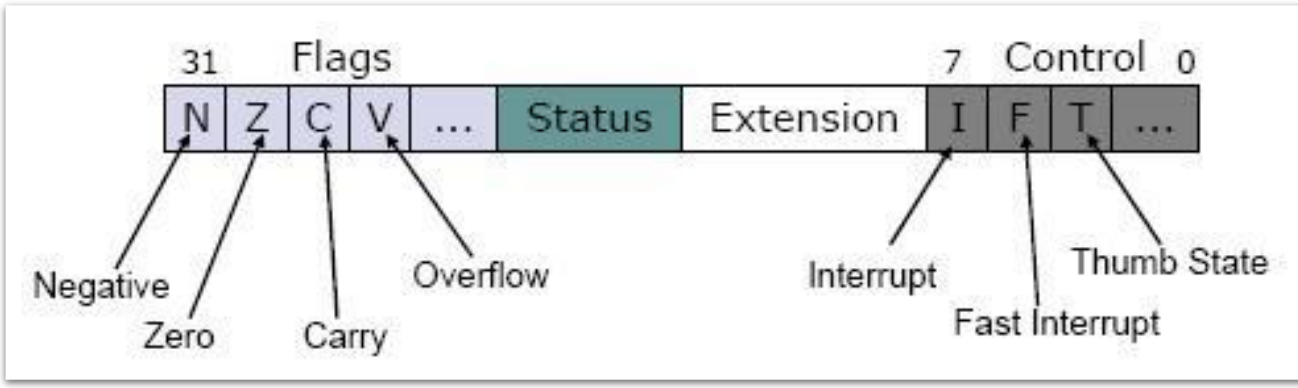


Архітектура HARVARD.

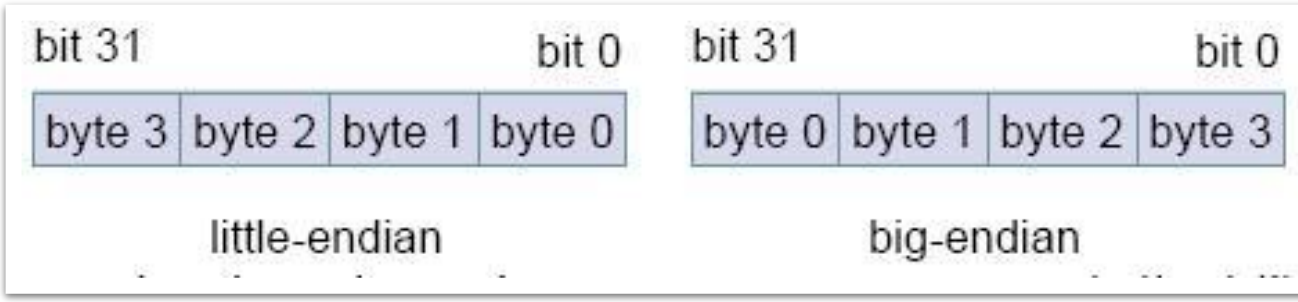


Регістри доступні в режимі користувача USER MODE.





ARM використовує 32-бітові адреси.



2.2 Звід інструкцій

Переміщення даних.

ARM має архітектуру завантаження-збереження[1].

Дані повинні бути завантажені в регістри, перш ніж вони можуть бути оброблені ALU.

Дані переміщуються між регістрами шляхом застосування інструкції

MOV r1, r2 ; r1 = r2

MOV r3, #1 ; r3 = 1

Дані переміщуються між модулями за допомогою інструкцій LOAD та STORE.

Базова інструкція LOAD це LDR (завантаження слова в регістр), але є варіації, які застосовують байт[2], півслова[3] і байти із знаком[4].

Базова інструкція STORE це STR (зберегти слово з регістру), варіації STRB і STRH.

Приклад завантаження.

[1] Load-Store architecture
[2] byte, LDRB
[3] Half word, LDRH
[4] signed bytes, LDRSB

- Before:

r0 = 0x00000000

r1 = 0x00070000

mem32[0x00070000] = 0x00000005

LDR r0, [r1]

- After:

r0 = 0x00000005

r1 = 0x00070000

Інструкції для обробки даних.

- Інструкції обробки даних маніпулюють даними в регістрах (Move, Arithmetic, Logical, Comparison, Multiply).
- Якщо суфікс S використовується CPSR прапори N, Z, C, V оновлюється.
- ADD r1, r2, r3 does not update CPSR.
- ADDS r1, r2, r3 updates the CPSR.

Інструкції обробки даних та CPSR.

- MOVS r1, #1 ⇒ NZCV = 0000
- MOVS r2, #-1⇒ NZCV = 1000
- ADD r3, r2, r1⇒ NZCV = 1000
- ADDS r3, r2, r1⇒ NZCV = 0110

Інструкції обробки даних.

- Move: MOV, MVN
- Arithmetic: ADD, ADC, SUB, SBC, RSB,RSC
- Logical: AND, ORR, EOR, BIC
- Comparaison: CMP, CMN, TST, TEQ
- Multiply: MUL, MLA, SMLAL, SMULL, UMULL, SMLAL, UMLAL

Формати інструкцій обробки даних.

- Basic format
- SUB r3, r2, r1 ; r3 = r2 – r1
- Immediate Operand
- SUB r3, r2, #3 ; r3 = r2 - 3
- Preprocessing (Barrel-Shifter)
- SUB r3, r2, r1, LSL #1 ; r3 = r2 – (r1 * 2)

Схема циклічного зсуву.

Схема циклічного зсуву дозволяє введення зсуву до вступу даних в ALU.

Shift Operations: LSL, LSR, ASR, ROR, RRX.

Example:

MOV r3, r4, LSL #3; r3 = 8 * r4

Інструкції розгалуження.

- Інструкції розгалуження використовується для зміни потоку виконання (if-then-else, for-loop, while-loop)

- Відділення Інструкції: B, BL, BX, BLX
- Інструкції розгалуження часто використовуються з умовами (EQ, NE, CS, CC, MI, PL, VS, HI, LS, GE, LT, GT, LE)

BEQ label ; Branch to label, if Z = 1

- Адресний ярлик зберігається в інструкції, як PC-відносний зсув і повинно бути в межах 32 Мб від інструкції розгалуження

Subroutines.

- The BL (Branch and Link) instruction can be used for subroutines, since it writes the return address to the link register
- BL subroutine
- ...
- subroutine
- ... ; code for subroutine
- MOV pc, lr ; return by moving lr to pc

Умовне виконання.

- Not only branch instructions can be used with conditions
- ADDEQ r4, r5, r6 is only executed if Z = 1
- Conditional Execution helps to design shorter programs that do not use so much memory

Висновки до лекції 2.

- ARM є сім'я ядер мікропроцесорів.
- Архітектура завантаження / збереження.
- Більшість інструкцій RISC, працюють у єдиному технологічному циклі.
- Деякі операції для багатьох регістрів виконуються довше.
- Всі команди можуть бути виконані умовно.

3 ПРИНЦИП ДІЇ GPS

Agenda:

- 1) загальні відомості щодо GPS;
- 2) навігаційні сигнали.

3.1 Загальні відомості щодо GPS

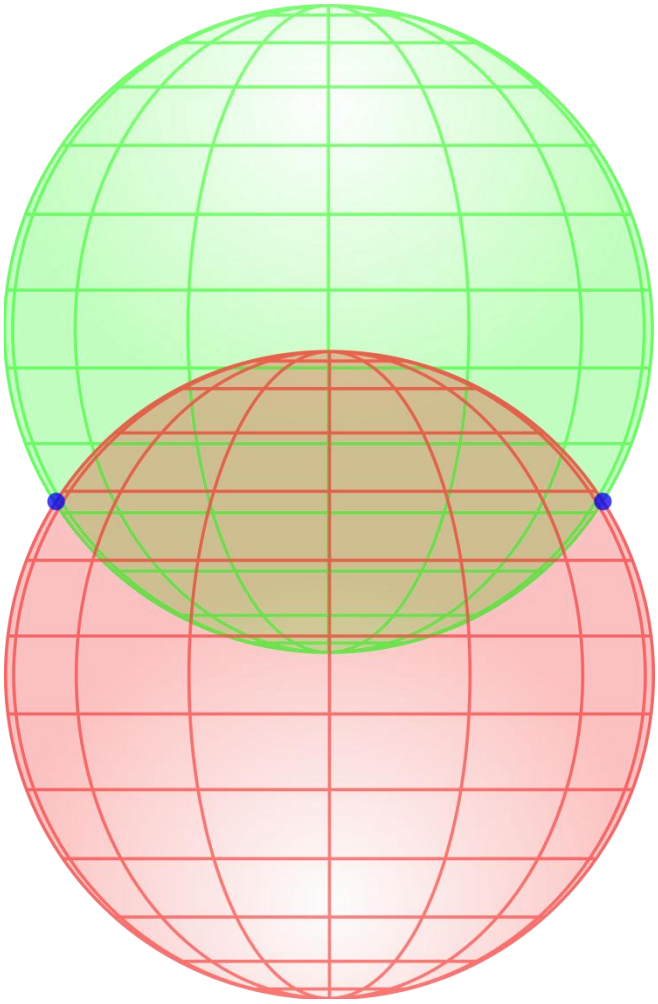
В 1957році Радянський Союз запустив перший Супутник. Команда американських вчених під проводом доктора Ричарда Кешнера (Richard B. Kershner) здійснювала моніторинги радіопередач Супутника. Вони виявили, що із-за ефекту Допплера , частота сигналу Супутника був вища, коли супутник наближався, і нижче, коли він віддалявся від них. Вони усвідомили, що з тих пір, як вони знали своє точне місцеположення на глобусі, вони могли б точно визначити, де супутник був уздовж своєї орбіти, змірявши частоту Допплера.

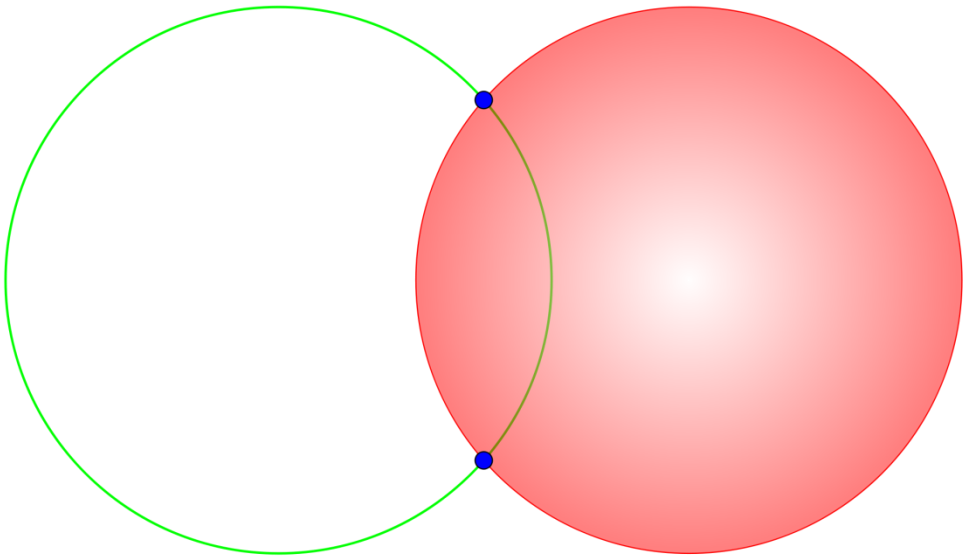
Концепція GPS полягає у наступному.

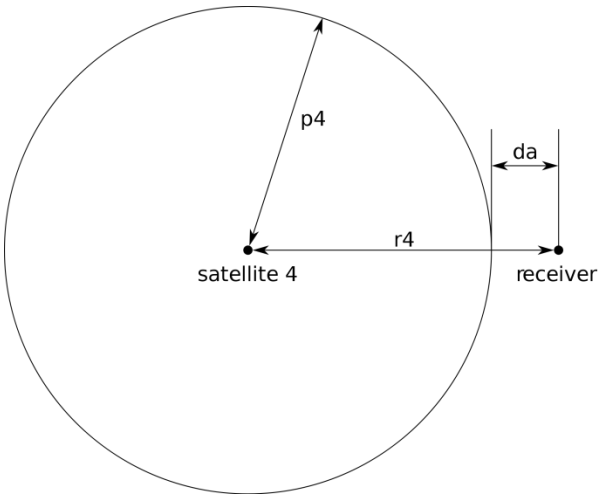
Кожен супутник безупинно передає повідомлення, що містять:

- час, коли повідомлення було послано,
- параметри для розрахунку точної орбіти для супутника(ephemeris),
- а також данні щодо загальносистемного здоров'я і грубих орбіт всіх супутників GPS (альманах).

Принцип визначення розташування користувача наведено на рисунках.







Розв’язання навігаційної задачі наведено нижче.

Приймач використовує повідомлення, отримані з супутників, щоб визначити позиції супутників і час відправлення. Компоненти x,y, z положення супутників і час відправлення позначені як [x_i, y_i, z_i, t_i], де індекс і позначає номер супутника і має значення 1, 2, ..., n. Знаючи, коли повідомлення було отримано t_r, приймач обчислює час проходження повідомлення як (t_r - t_i). Припускаючи, що повідомлення подорожувало зі швидкістю світла (c) відстань становить (tr - ti)c. Знання відстані від приймача до супутника і положення супутника означає, що приймач знаходиться на поверхні сфери з центром у позиції супутника. Таким чином, приймач на або поблизу точки перетину поверхні сфер. В ідеальному випадку відсутності помилок, приймач на перетині поверхні сфер. Нехай b похибка годинника. Приймач має чотири невідомих, три компонента GPS положення приймача і зміщення годинника [x, y, z, b]. Рівняння поверхні сфери визначається за формулою:

$$(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 = \left([t_r + b - t_i]c\right)^2,\ i = 1, 2, \ldots, n$$

або в термінах дальності

$$p_i = (t_r - t_i)\, c,$$

як

$$p_i = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2} - bc,\ i = 1, 2, ..., n$$

Три коди PRN для вимірювання дальності, передаються:

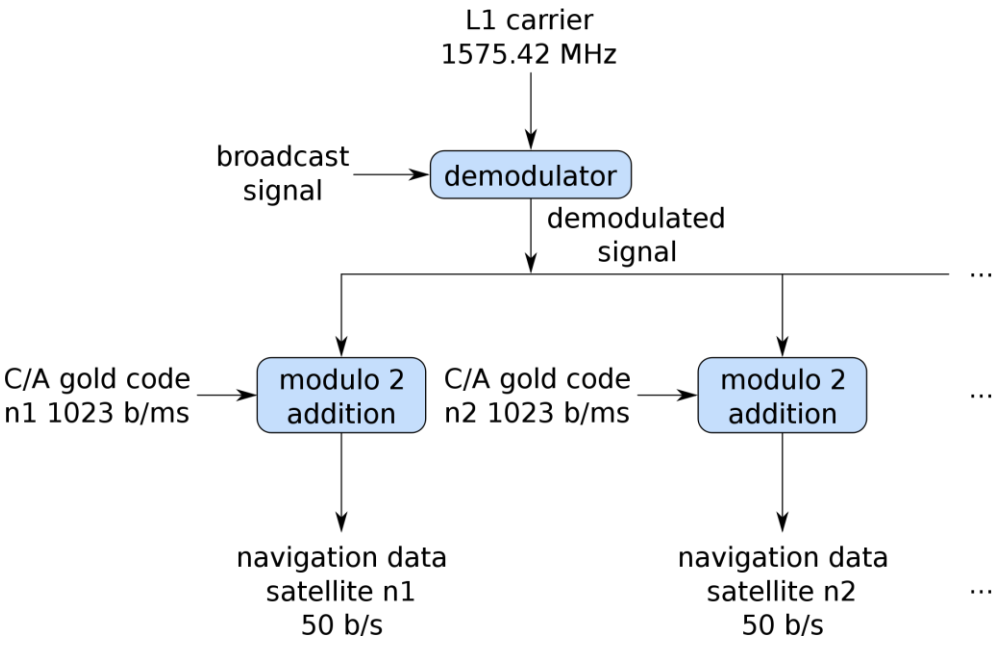
точний (P) код, який є головним кодом для вимірювання відстані;

Y-код, який використовується замість P-коду кожного разу, коли A-S метод дії активізований;

C/A[5] код, який використовується перш за все для пошуку коду P (або Y) (означають як P(Y)).

[5] coarse/acquisition, C/A

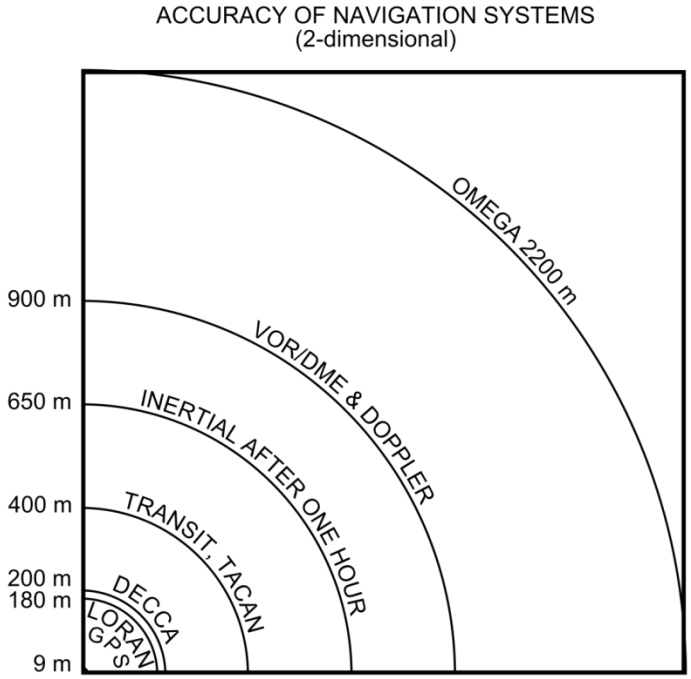
Для декодування C/A коду може застосовуватися така схема



Станом на 2010 рік космічна угрупування нараховує 31 супутник.

Block	Launch Period	Satellite launches				Currently in orbit and healthy
		Success	Fail	In preparation	Planned	
I	1978–1985	3	1	0	0	0
II	1985–1990	2	0	0	0	0
IIA	1990–1997	12	0	0	0	11 of12
IIR	1997–2004	12	1	0	0	12of13
IIR-M	2005–2009	8	0	0	0	7of8
IIF	2010–2011	1	0	11	0	1of1
IIIA	2014-?	0	0	0	12	0
IIIB		0	0	0	8	0
IIIc		0	0	0	16	0
Total		52	2	11	36	30
(Last update: 24 May 2010)						

Порівняльна характеристика різних систем наведена нижче.



3.2 Навігаційні сигнали

Наведений нижче документ конкретизує технічні вимоги до GPS і ніякі терміни з наведених в ньому не можуть бути змінені на альтернативні для будь-якого виконання договору або в замовлені на постачання між усіма учасниками проекту.

UNCLASSIFIED

REVISIONS	
DESCRIPTION	DATE
ICD-GPS-200, Revision C, Initial Release	10 October 1993
IRN-200C-001	13 October 1995
IRN-200C-002	25 September 1997

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED.

APPROVALS

AUTHORIZED SIGNATURES	REPRESENTING	DATE
Signature on file	GPS NAVSTAR JPO SMC/CZ (AFMC)	15 December 1994
Signature on file	ROCKWELL INTERNATIONAL SPACE SYSTEMS DIVISION	16 November 1993
Signature on file	ROCKWELL INTERNATIONAL COLLINS AVIONICS & COMMUNICATIONS DIVISION	15 November 1993
Signature on file	INTERNATIONAL BUSINESS MACHINES (IBM) FEDERAL SYSTEMS COMPANY	02 December 1993
Signature on file *	MARTIN-MARIETTA ASTRO SPACE DIVISION	05 August 1994

* An asterisk affixed to the approval signature indicates that the approval is subject to exceptions taken in the "Letter of Exception" contained in Appendix I of this document.

INTERFACE CONTROL DOCUMENT 10 Oct 1993

UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES.
TOLERANCES ON:
DECIMALS ANGLES
XX = ±0.03 ±0° 30'
XXX = ±0.01

DR BY
CHK BY
APPROVALS

ARINC RESEARCH CORPORATION
2250 E. Imperial Highway, Suite 450
El Segundo, CA 90245-3509

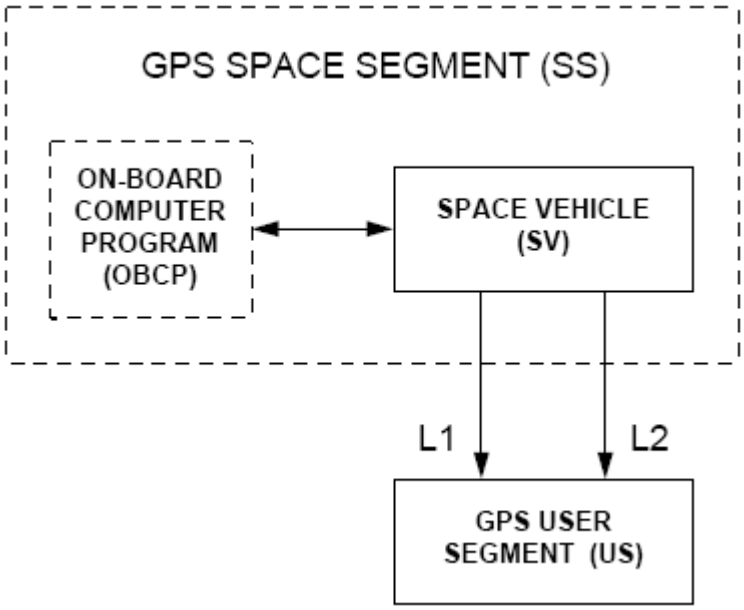
ICD TITLE
Navstar GPS Space Segment / Navigation User Interfaces

THIS DOCUMENT SPECIFIES TECHNICAL REQUIREMENTS AND NOTHING HEREIN CONTAINED SHALL BE DEEMED TO ALTER THE TERMS OF ANY CONTRACT OR PURCHASE ORDER BETWEEN ALL PARTIES AFFECTED.

SIZE	CODE IDENT NO.	DRAWING NO.
A	OVYX1	ICD-GPS-200
SCALE: N/A	REV: C	SHEET 1

UNCLASSIFIED

Документ визначає інтерфейс між двома сегментами.



Несучі сигнали каналів зв'язку, які працюють в L-діапазоні, модулюються двома бінарними послідовностями, кожна з яких нормально є композит, що виробляється додаванням за модулем 2:

псевдо випадкової шумової послідовності (PRN) для вимірювання відстані;

передаванні з космосу дані системи (тобто навігаційні дані NAV data).

Данні передаються фреймами тривалістю 30 секунд, які мають у своєму складі 5 часткових фреймів(Subframes=6s).

Subframes	Description
1	Satellite clock, GPS time relationship
2–3	Ephemeris(precise satellite orbit)
4–5	Almanac component (satellite network synopsys ,error correction)

Формування фреймів (повідомлення HOW[6]).

[6] Hand-Over Word

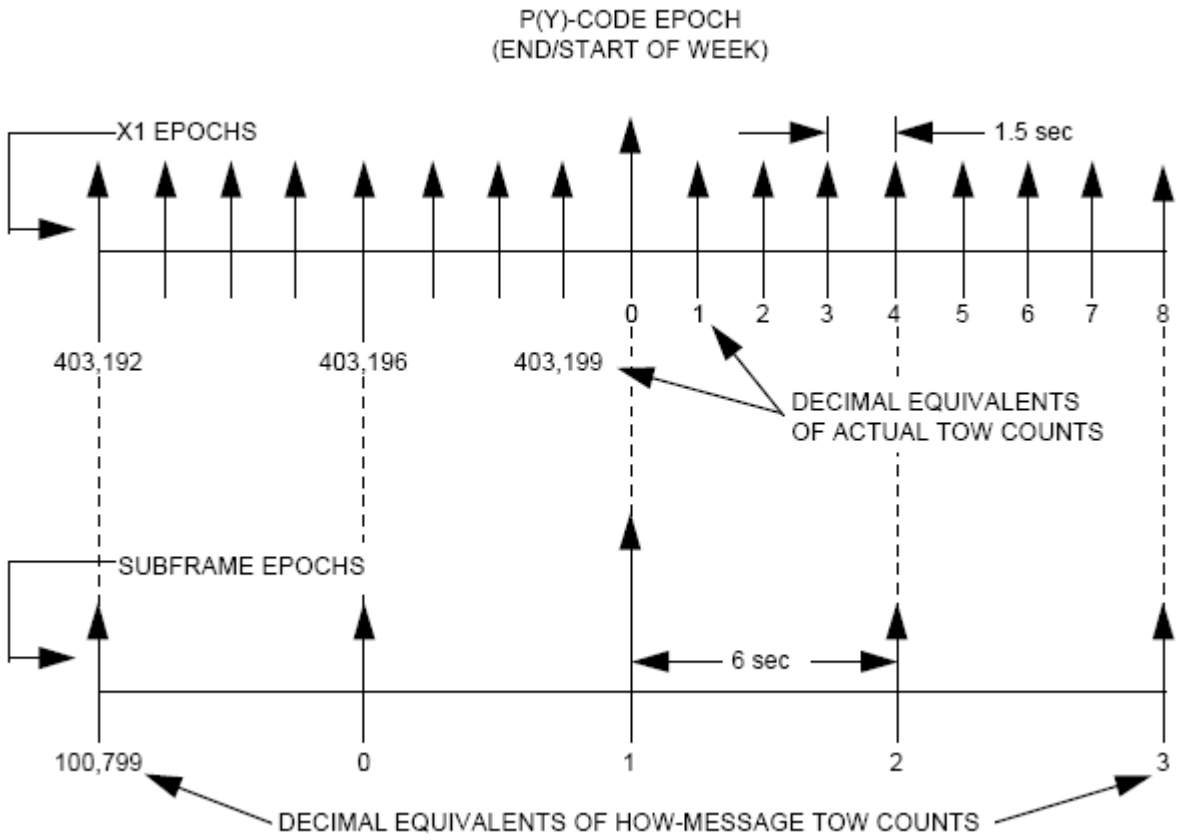
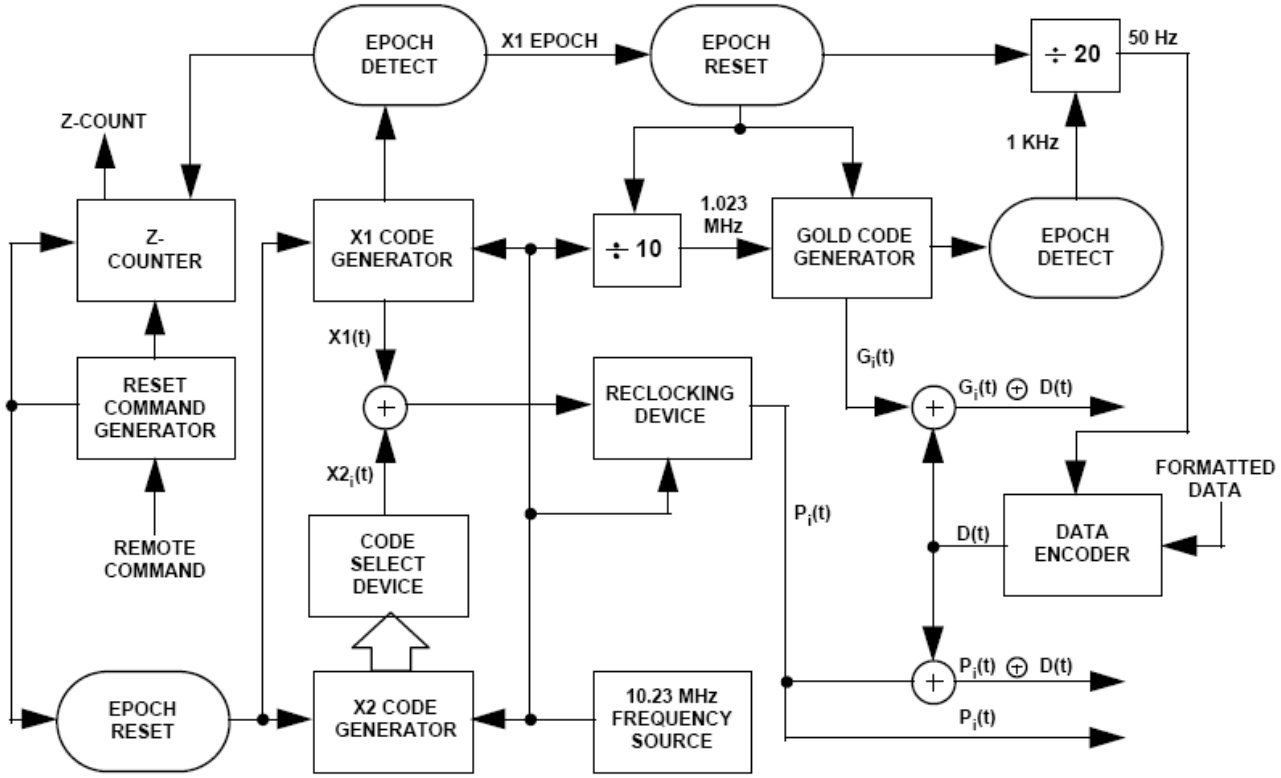
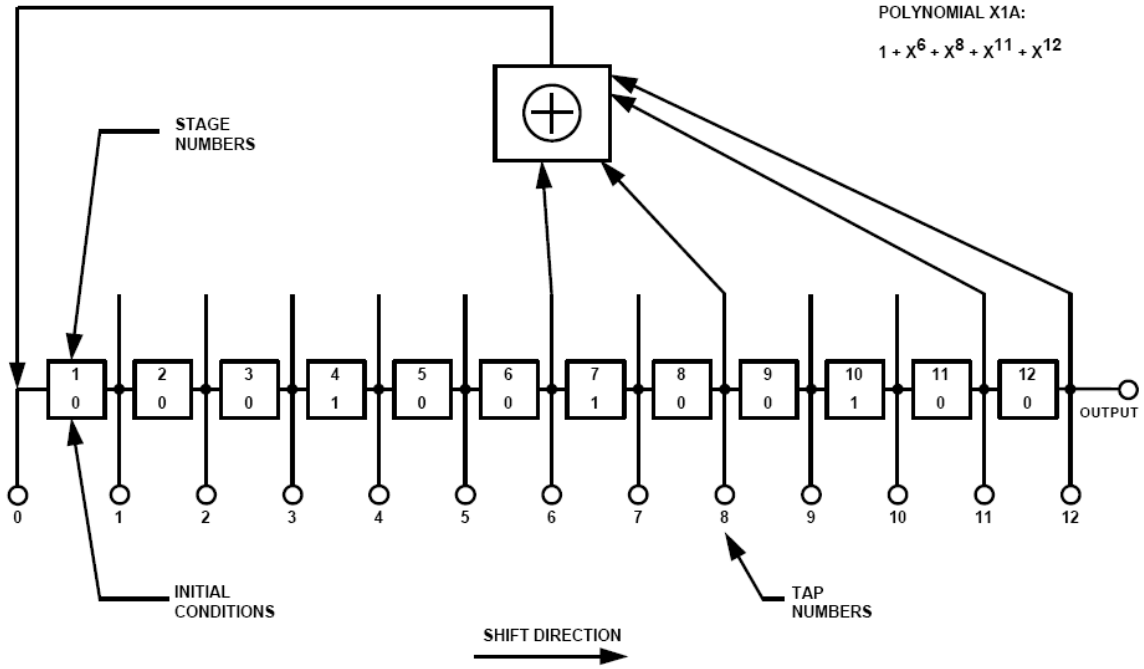


Схема генерації псевдовипадкових шумових послідовностей (PRN) та модуляції сигналів наведена нижче.



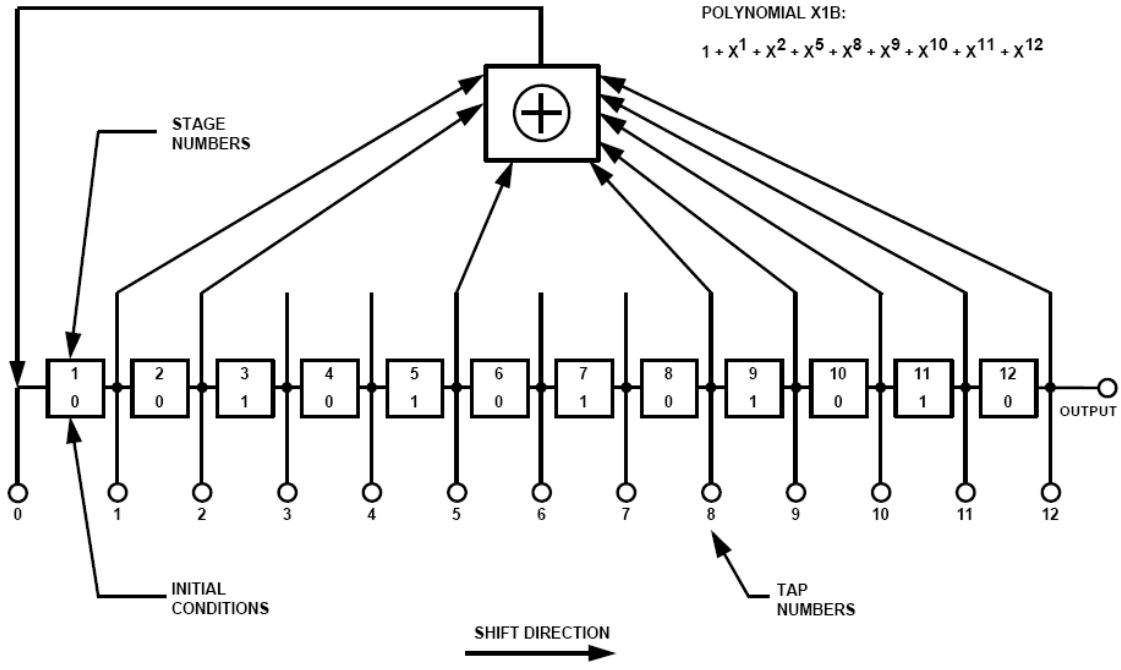
Для і-го супутника Р код - це сума за модулем 2 двох псевдо випадкових послідовностей X1(t) і X2(t - iT), де Т і є періодом одного чипа і дорівнює 1/(1.023 x 3,000,000) секунди, тоді як і - є ціле число від 1 до 37.

Послідовність X1 – це сума за модулем 2 двох псевдо випадкових послідовностей X1A та X1B, які формуються 12-розрядними регістрами , період цих послідовностей обмежується значеннями 4022 та 4023 відповідно.

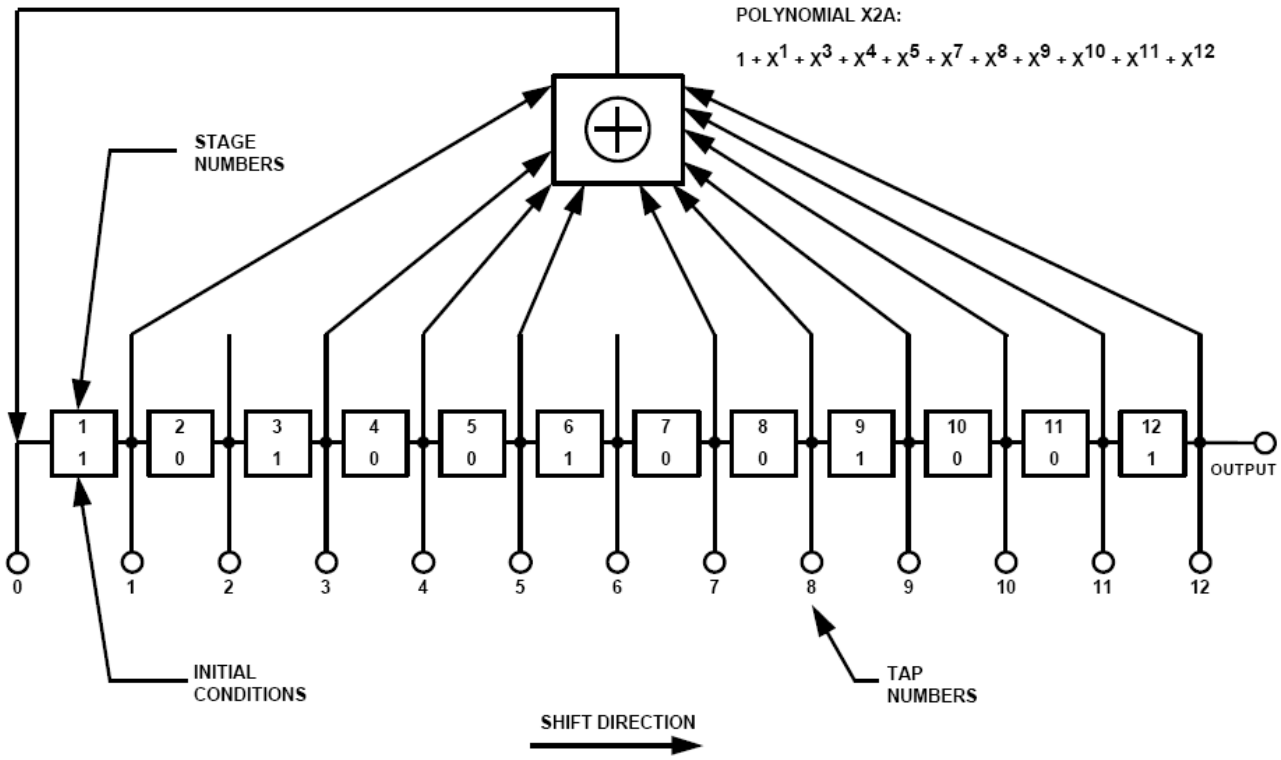


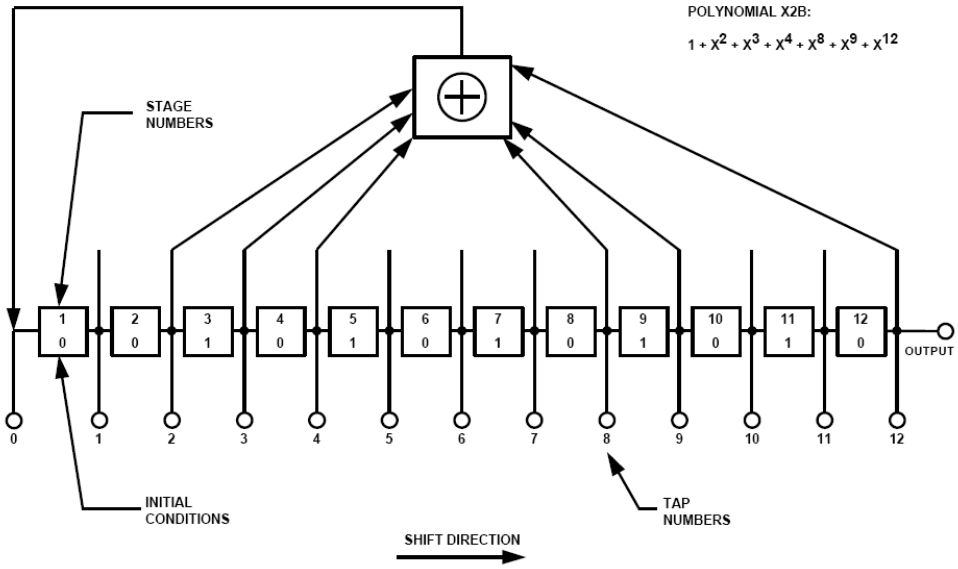
Коли буде сформовано 3750 періодів послідовності X1A, формується номер епохи X1.

Епоха X1 відбувається кожні 1,5 секунди після того як 15,345,000 чипів послідовність X1 сформовано.

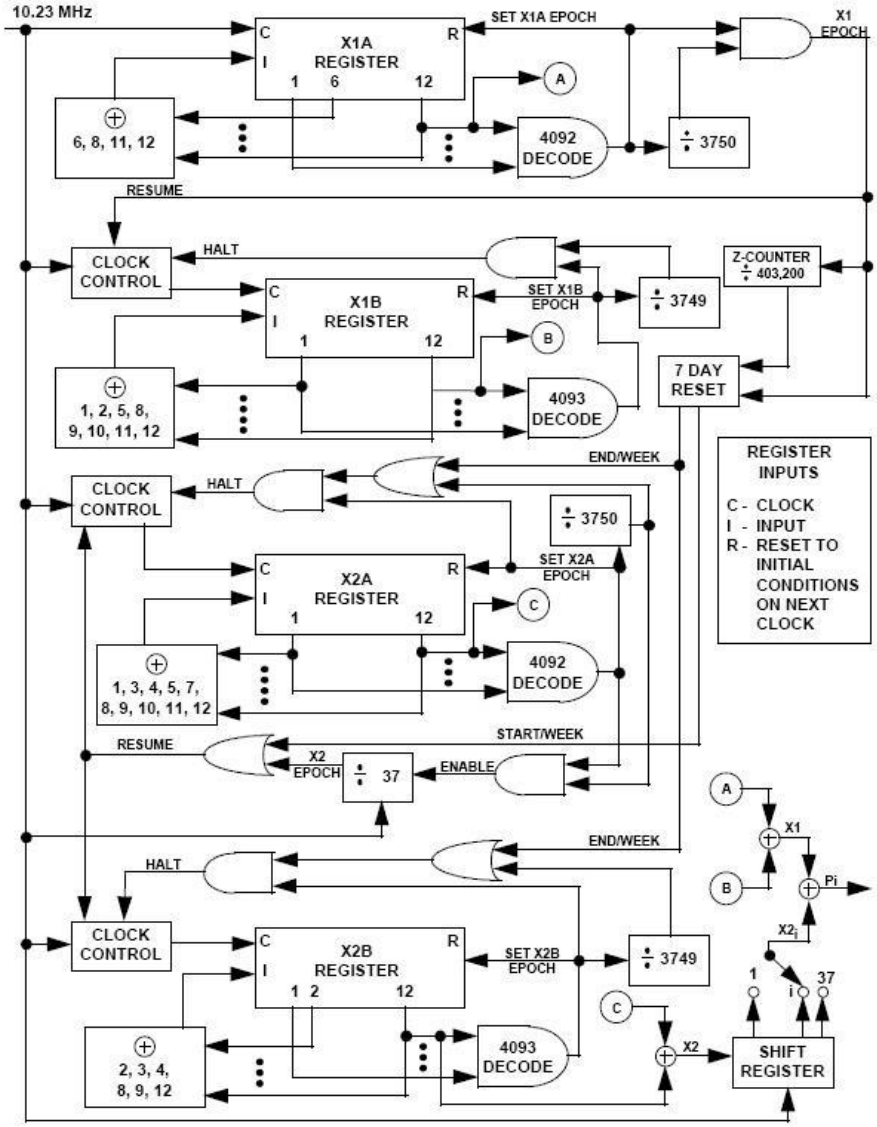


Епоха X1 відбувається кожні 1,5 секунди після того як 15,345,000 чипів послідовність X1 сформовано.

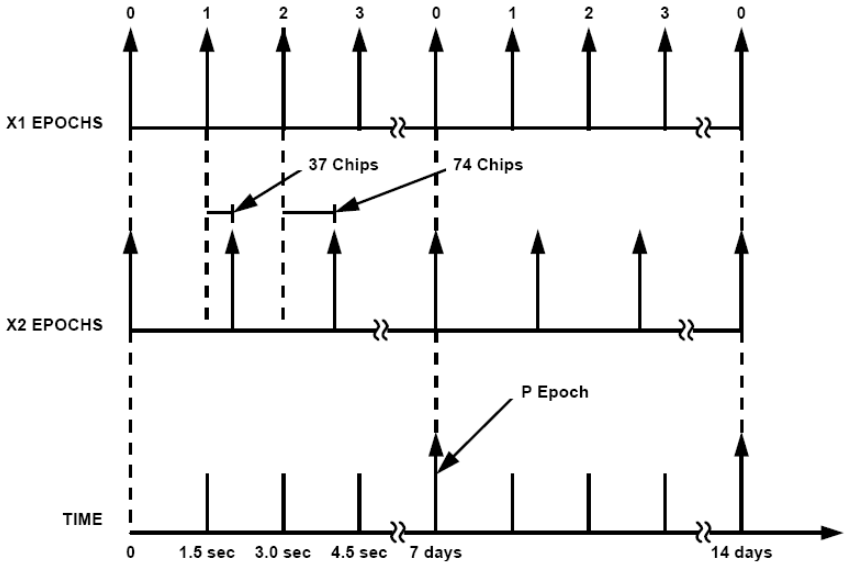




Генерація P коду відбувається за схемою.



Часові співвідношення P коду наведені на часовій діаграмі.



Останні 400 мікросекунд тижня

Table 3-IV. P-Code Reset Timing

(Last 400 μsec of 7-day period)

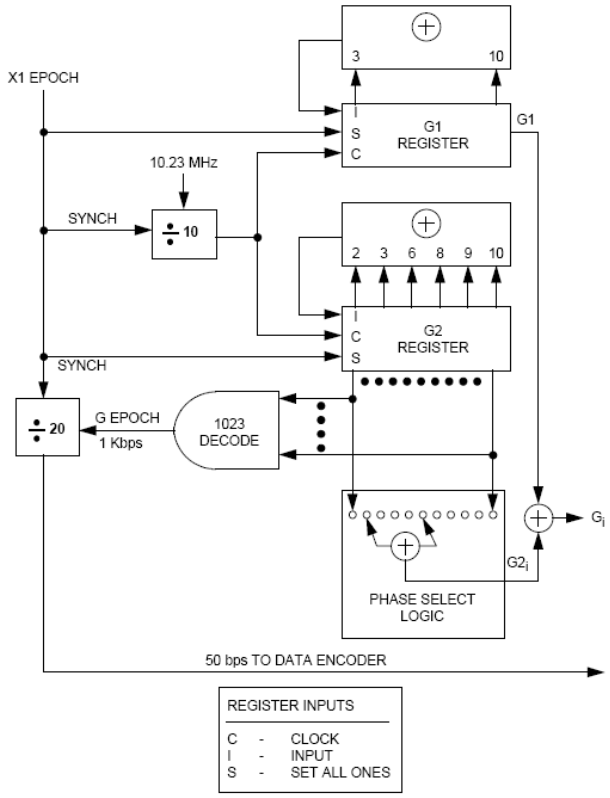
	Code Chip			
	X1A-Code	X1B-Code	X2A-Code	X2B-Code
TIME ↓	1	345	1070	967
	•	•	•	•
	•	•	•	•
	•	•	•	•
	3020	3367	4092	3989
	•	•	•	•
	•	•	•	•
	•	•	•	•
	3127	3471	4092	4093
	•	•	•	•
	•	•	•	•
	•	•	•	•
	3749	4093	4092	4093
	•	•	•	•
	•	•	•	•
	•	•	•	•
	4092*	4093	4092	4093

* Last Chip of Week.

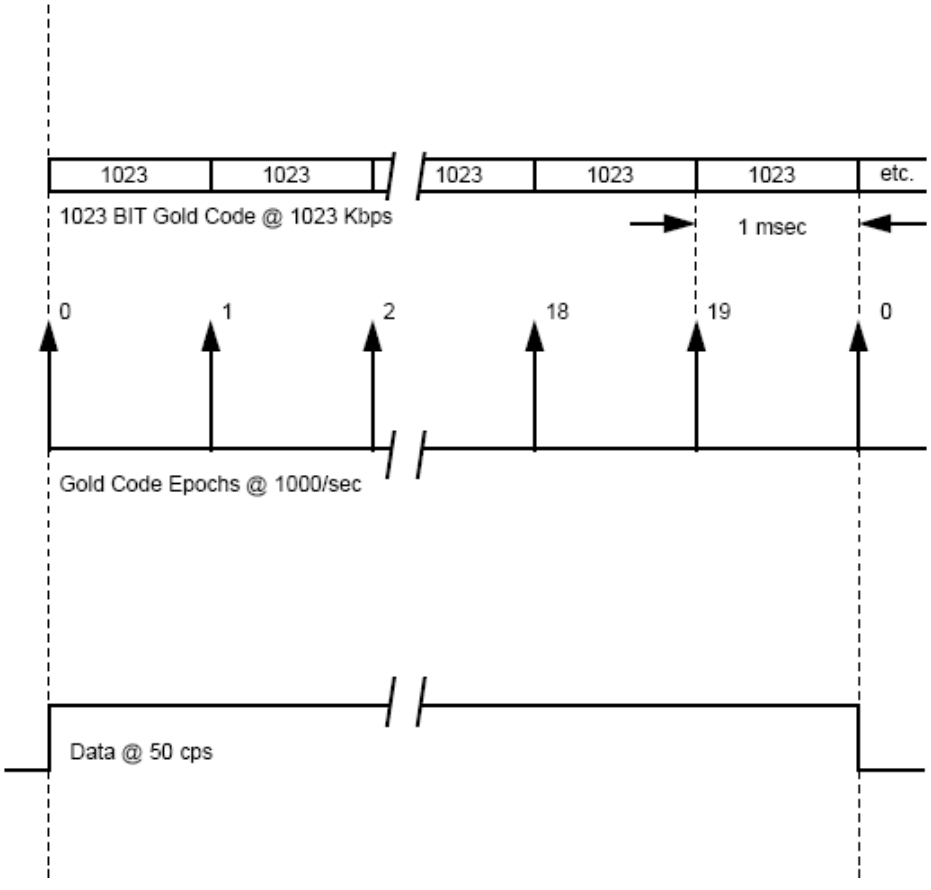
Вектори станів генераторів.

Code	Chip Number	Vector State	Vector State for 1st Chip following Epoch
X1A	4021	30030303	003030300
	4022	000303030	
X1B	4022	3033331	0333330
	4023	0033333	
X2A	4021	113030301	30303031
	4022	13030303	
X2B	4022	00033331	0333330
	4023	0033333	

Схема С/А генератору наведена нижче.



Синхронізація епох наведена нижче.

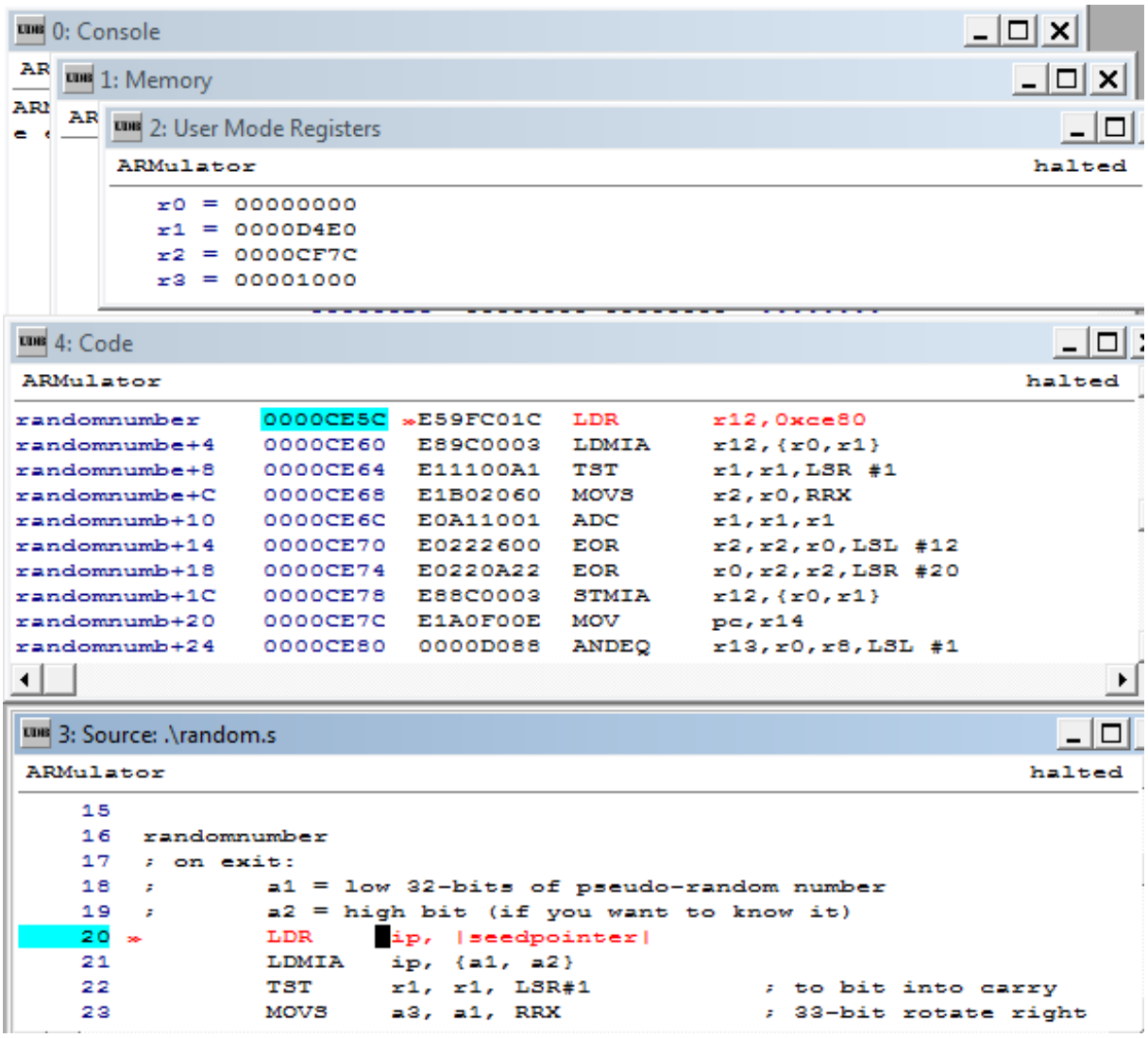


Тематичний модуль 2. Операційна система

Лекція 4. Контекстна схема

Лекція 5. Процес обробки переривань

Лекція 6. Процес припинення задачі



4 КОНТЕКСТНА СХЕМА

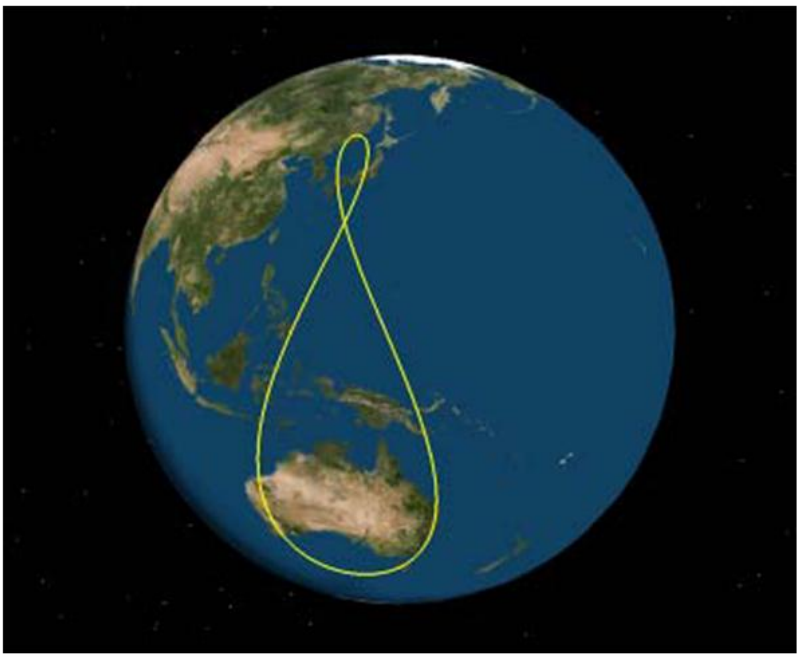
Agenda:

- 1) дві форми операційних систем;
- 2) контекстна схема операційної системи.

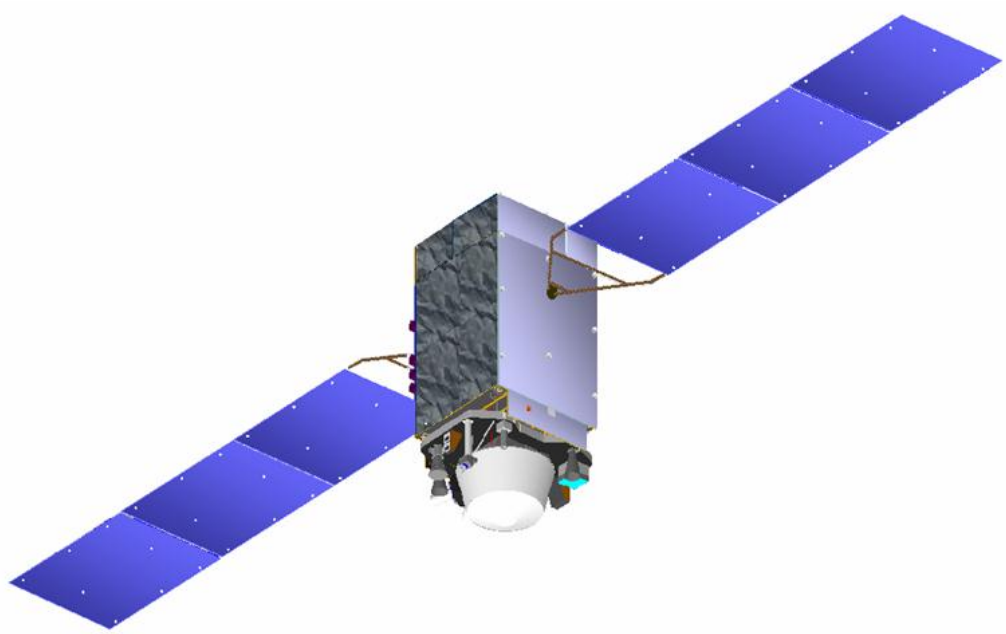
4.1 Дві форми операційних систем

9/9/2011 СИСТЕМИ СУПУТНИКОВОГО ЗВ'ЯЗКУ І НАВІГАЦІЇ 3

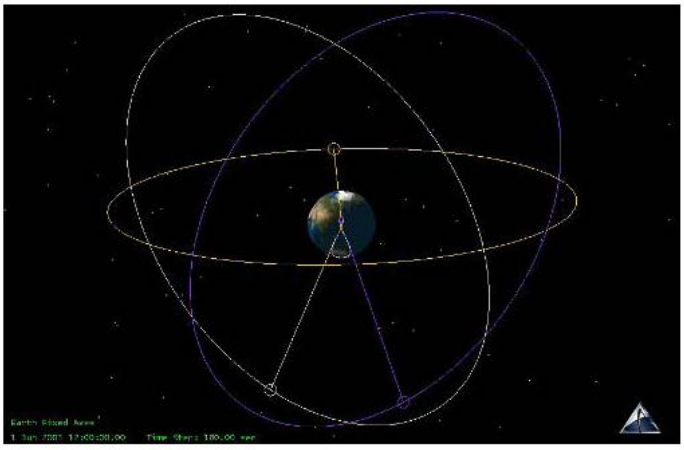
Трек від QZSS



Супутник QZSS

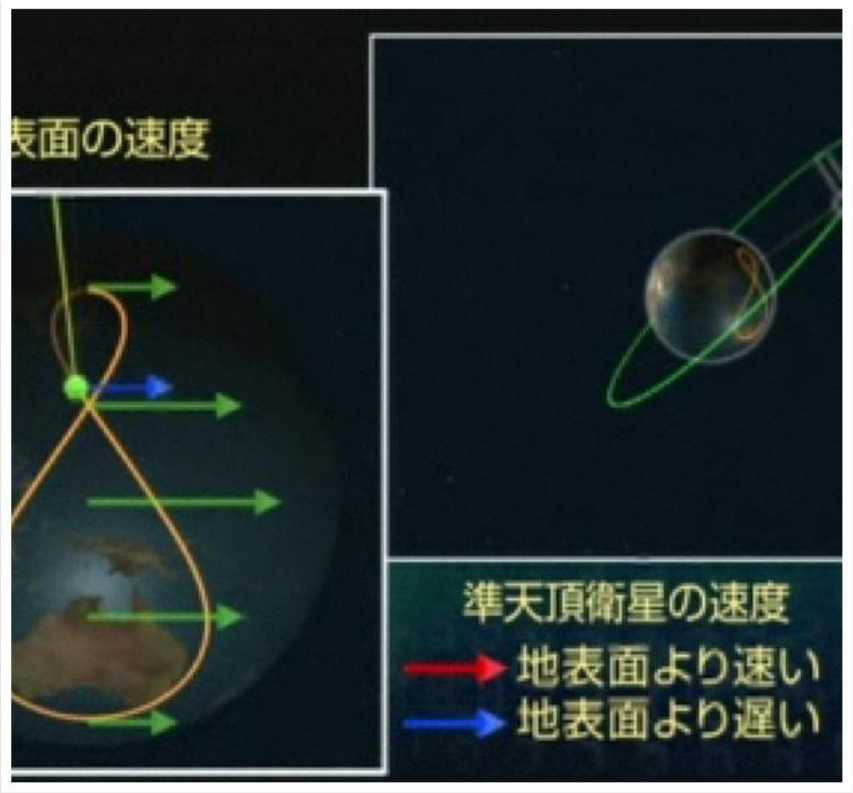


Орбіти QZSS



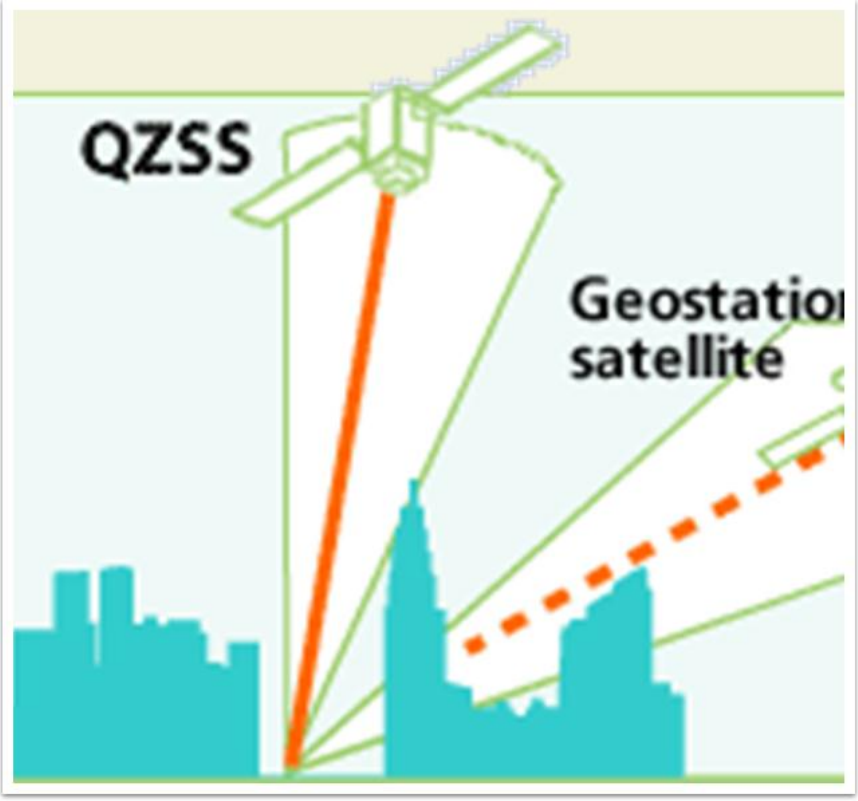
Орбіти QZSS

http://www.jaxa.jp/countdown/f18/overview/orbit_e.html



Орбіти QZSS

http://www.jaxa.jp/countdown/f18/overview/orbit_e.html



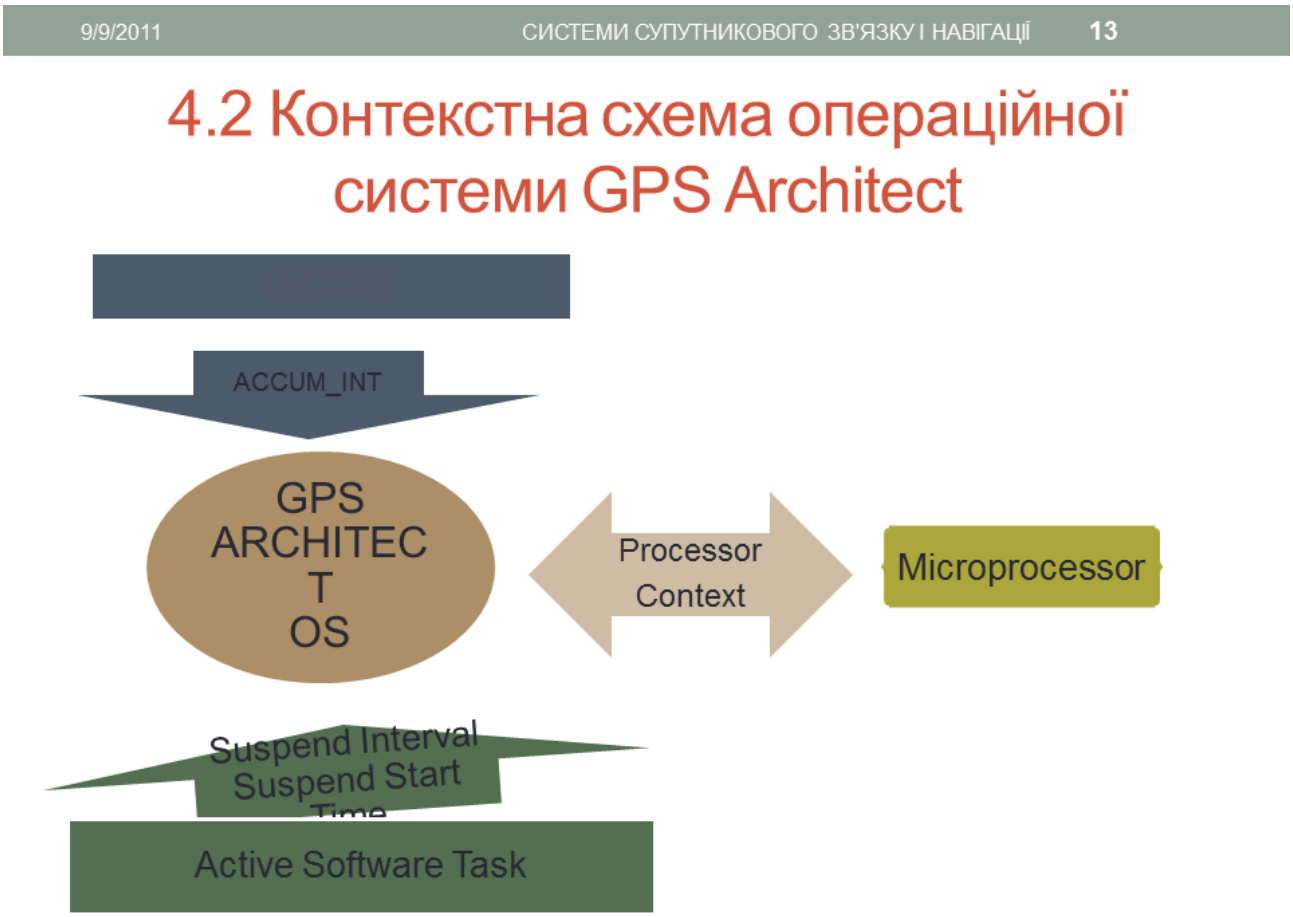
4.1 Послідовна операційна система

Переваги	Недоліки
Проста структура для послідовних задач	Не влаштовує виконання операцій у реальному часі
Легше виконувати завантаження мікропроцесора	Не влаштовує задачі з різними пріоритетами
Мінімальні вимоги до пам’яті для збереження контекстів задач	Відсутня гнучкість при внесенні змін в програмну структуру
Легке повторне завантаження мікропроцесора	Не дозволяє легко вирішувати призначення пріоритетів
--	Важко повно використовувати потужність процесору

4.1 Паралельна операційна система

Переваги	Недоліки
Пасує до операцій у реальному часі	Необхідна пам’ять великої місткості для збереження контексту задачі
Пасує до задач з різними пріоритетами	Додаткове завантаження мікропроцесора для перемикання задач
Вирішує задачу надання пріоритетів	Велика та складна структура
Гнучка до додавання нових програмних задач	Важко передбачити навантаження мікропроцесора

4.2 Контекстна схема операційної системи



4.2 Ініціалізація масиву TCB структур

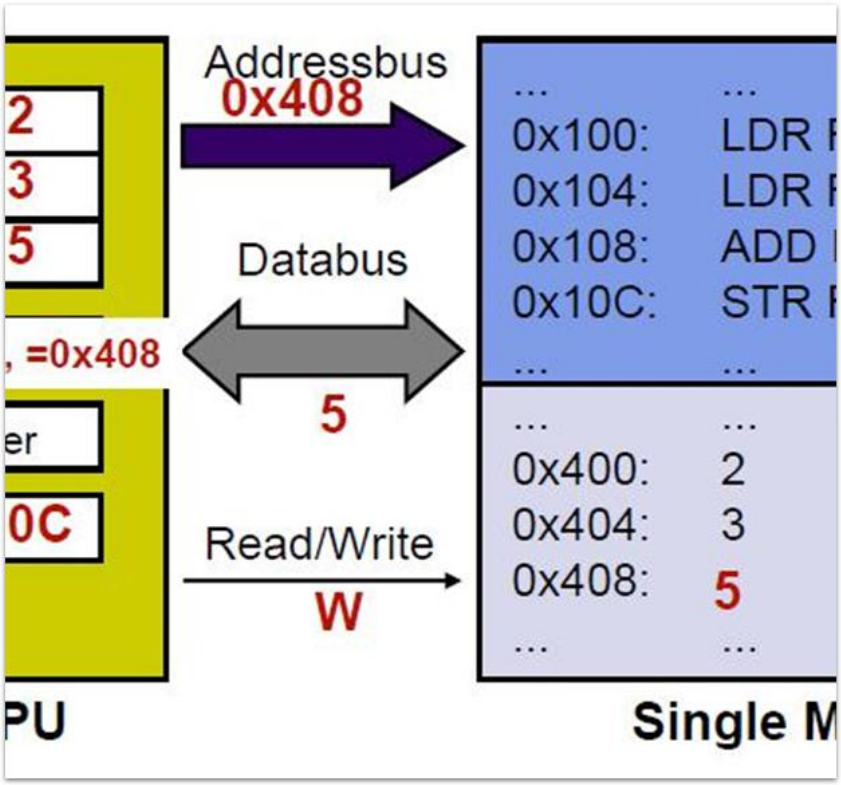
```
tcbstruc TCB[] =
{
    {TTakeMeas,NULL,0,"TTakeMeas",2000},
    {TNav,NULL,0,"TNav",5000},
    {TDisplay,NULL,0,"TDisplay",9000},
    {TRTCM,NULL,0,"TRTCM",2000},
    {TProcSbf,NULL,0,"TProcSbf",2000},
    {TAlloc,NULL,0,"TAlloc",9000},
    {NULL,NULL,0,"MAIN",0}                /* MAIN must be
at the end. */
};
```


4.2 Оригінальне визначення стеку

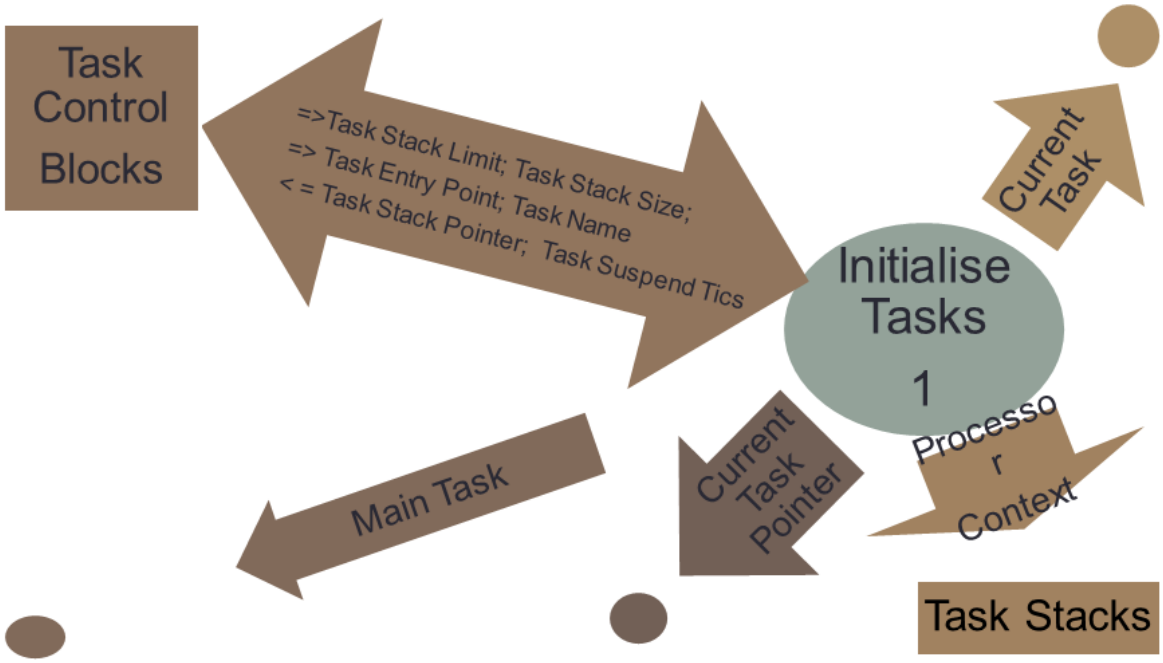
Покажчик стеку	Регістр	Значення
База стеку	r0..r9	Don’t care
База стеку – 0x28	r10(sl)	Ліміт стеку задачі
База стеку – 0x2C	R11, r12	Don’t care
База стеку – 0x34	R13(sp)	Покажчик стеку задачі (оригінальне значення „ База стеку - 0x44)
База стеку – 0x38	R14	Don’t care
База стеку – 0x3C	R15(PC)	Точка входу в задачу
База стеку – 0x40	CPSR	Регістр статусу поточної програми

4.2 Розташування стеку

Стек розташований нижче області даних



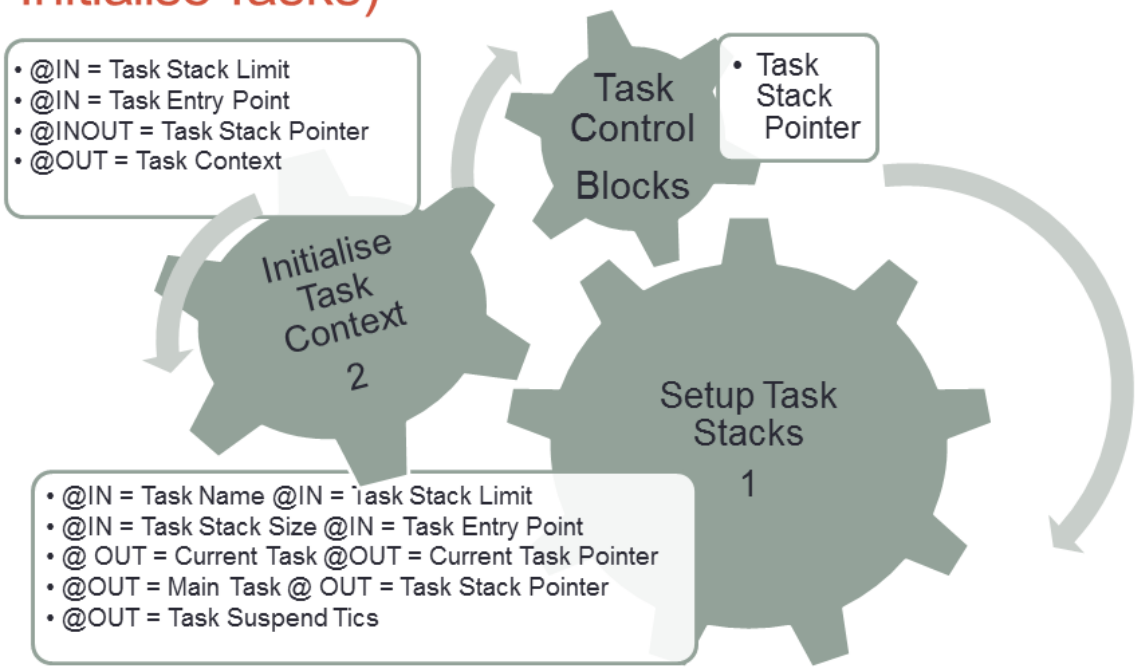
4.2 Схема потоків даних при ініціалізації задачі (Parent: GPS Architect OS)



4.2 Визначення типу структури

```
typedef struct
{
    /* The first 3 entries must be maintained and in this order. These
are
       referenced, with respect to the start address of the TCB, directly
       in the file ARMASM.S. */
    void (*TStart)(void);                   /* Task startup address. */
    void *TStackLimit;                  /* Pointer to task's stack limit. */
    unsigned long TStackPointer;            /* Stack pointer for task. */
    /* New entries can be added after this point. */
    char TName[10];             /* Task name string, up to 9 characters. */
    unsigned long TStackSize;       /* Size of stack to allocate (bytes).
*/
    unsigned short TSuspTics;      /* Incremental suspend interval
(TICs). */
} tcbstruc;
```


4.2 Схема потоків даних при формуванні стеків та ініціалізації контексту задачі (Parent: Initialise Tasks)



4.2 Специфікація процесу завдання стеків програмної задачі

- @IN = Task Name @IN = Task Stack Limit @IN = Task Stack Size @IN = Task Entry Point @OUT = Current Task @OUT = Current Task Pointer @OUT = Main Task @OUT = Task Stack Pointer @OUT = Task Suspend Tics
- @PSPEC Setup Task Stacks
- //For each task, (except MAIN), get a pointer to the stack limit and stack pointer
- task counter = 0;
- DO Task Stack Limit = memory allocation (Task Stack Size);
- Task Stack Pointer = Task Stack Limit + Task Stack Size task counter = task counter + 1
- WHILE (Task Name != "MAIN“)
- // Set the current task and current task pointer to be MAIN and MAIN TCB
- Current Task = Main Task = task counter; Current Task Pointer = Task Entry Point of Main Task
- MainTaskSuspendTics = 0 FOR task counter 0 TO MainTask -1 TaskSuspendTics = 0 Initialise Task Context() ENDFOR @

4.2 Специфікація процесу ініціалізації контексту програмної задачі

- @IN = Task Stack Limit @IN = Task Entry Point
- @INOUT = Task Stack Pointer
- @OUT = Task Context
- @PSPEC Initialize Task Context
- Save Task Context to Task Stack Pointer in descending stack:R0-r15,CPSR
- Where: R10 = Task Stack Limit R13 = Task Stack Pointer R15 = Task Entry Point Others registers = don’t care
- Update Task Stack Pointer with new stack pointer
- @

4.2 Коментар до програми

- #include "includes.h”
- /***
- * Function: void InitialiseTasks(void)
- * Revision: 6.02
- * Last Update: 09/16/2009
- * Allocate memory to the defined task stacks and initialise the TCB
- * structures for each task.
- * Input: None.
- * Output: None.
- * Return Value: None.
- ***/

4.2 Програма мовою С

- void InitialiseTasks(void)
- {int task_counter;
- /*Allocate stacks for all the tasks (except MAIN, which already has a stack and is currently running). */ task_counter = 0; while(strcmp(TCB[task_counter].TName,"MAIN"))
- { /* Allocate the memory. */
- TCB[task_counter].TStackLimit = (void*)malloc((size_t)(TCB[task_counter].TStackSize+32));
- /* Halt if memory allocation error. */ if(TCB[task_counter].TStackLimit==(void *)0) {#ifdef DEBUG printf("\nmalloc() failure for task number %d\n",task_counter); #endif while(1); }
- /* Stack pointer starts at highest address - descending stack. */
- TCB[task_counter].TStackPointer=(unsigned)((int)TCB[task_counter].TStackLimit +TCB[task_counter].TStackSize); task_counter++; }
- /* The current task at startup is the MAIN task. */ CurrentTask = MainTask = task_counter;
- /* Suspend TICs for MAIN is always zero - it can't be suspended. */
- TCB[MainTask].TSuspTics = 0; CurrentTaskPointer = &TCB[MainTask];
- /* Place all the other tasks on the suspended list with zero suspend and initialise the task contexts. */ for(task_counter=0;task_counter<MainTask;task_counter++) { TCB[task_counter].TSuspTics = 0; InitialiseContext(&TCB[task_counter]); }
- }

5 ОБРОБКА ПЕРЕРИВАНЬ

Agenda:

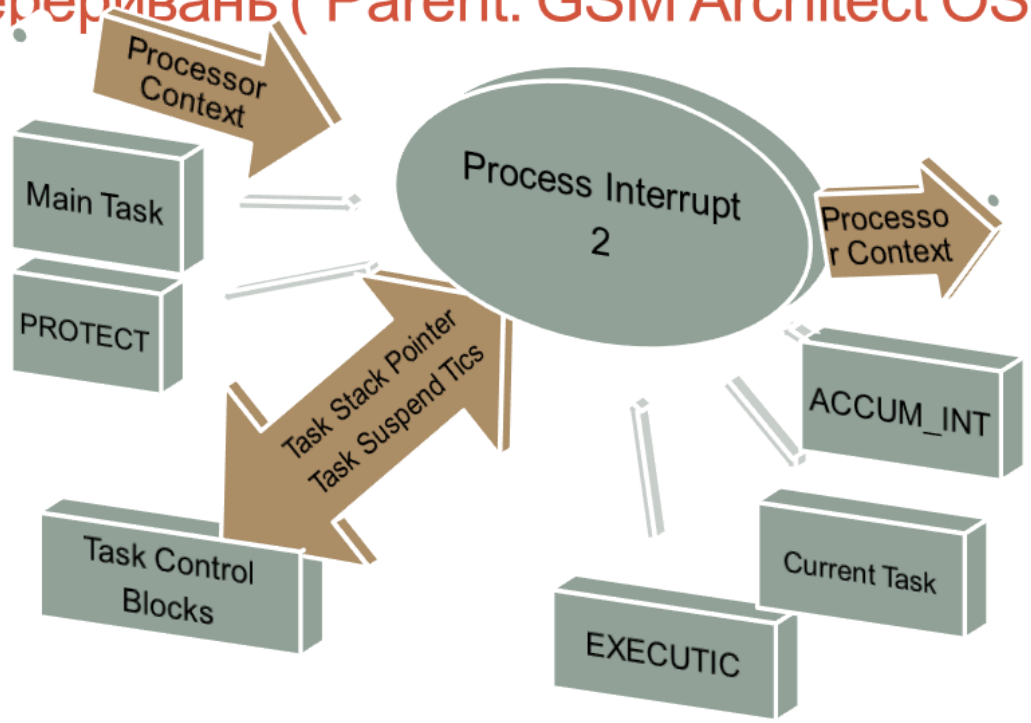
- схеми потоків даних;
- специфікації процесів.

5.1 Схеми потоків даних

5.1 Обробка переривань

- Програма обробки переривань (Interrupt Service Routine – ISR) зберігає контекст поточної задачі, якою управляє мікропроцесор.
- Коли надходить TIC зменшуються на один інтервали тимчасової зупинки задач та поновлюється виконання з найвищим пріоритетом.
- Коли TIC не надходить поновлюється задача, яка була збережена до входу в ISR.
- Коли TIC надійшов перемикач задачі може змінитися.

5.1 Схема потоків даних при обробці переривань (Parent: GSM Architect OS)



5.1 Схема потоків даних процесів 1,2,3 (Parent: Process Interrupt)

5.2 Специфікації процесів

5.2 Специфікація процесу збереження контексту програмної задачі

- @IN = ACCUM_INT
- @IN = Current Task Pointer
- @IN = Processor Context
- @OUT = Task Stack Pointer
- @OUT = Task Context
- @PSPEC Save Task Context
- Upon an ACCUM_INT(the microprocessor is in IRQ mode).
- Store the User Mode Processor Context (the interrupt task Task Context)
- R0-r14, SPSR&pc
- to the stack of the interrupted task.
- Store the User Mode Stack Pointer
- (the new Task Stack Pointer of the interrupted task)
- to the TCB of the interrupted task (pointed to by Current Task Pointer)
- @

5.2 Специфікація процесу зміни перемикача програмної задачі

- @IN = PROTECT @IN = Main Task Point
- @INOUT = Task Suspend Tics @INOUT = EXECUTIC
- @OUT = Current Task @OUT = Current Task Pointer
- @PSPEC Update Task Switcher
- //If task switching is allowed and TICs have occurred.
- IF PROTECT==0 and EXECUTIC!=0
- //Decrement the suspend intervals for all tasks.
- FOR Task = 0 TO Main Task -1 IF Suspend Tics for task > EXECUTIC
- Task Suspend Tics for task -= EXECUTIC
- ELSE Task Suspend Tics for task = 0 ENDIF ENDFOR
- //Find the highest priority task with zero suspend interval.
- FOR Task=0 TO Main Task IF Task Suspend Tics for task == 0
- Current Task = Task break ENDIF ENDFOR
- // The Current Task Pointer points to the new (or current) task.
- Current Task Pointer = Pointer to TCB of Current Task EXECUTIC = 0; ENDIF
- @

5.2 Специфікація процесу поновлення контексту програмної задачі

- @IN = Current Task Pointer @IN = Task Stack Pointer @IN = Task Context
- @OUT = Processor Context
- @PSPEC Restore Task Context
- The microprocessor is in IRQ mode.
- Restore the User Mode Stack Pointer (sp of the restored task) from the TCB Task Stack Pointer for restored task (pointed to by Current Task Pointer)
- Restore the User Mode Processor Context (the interrupt task Task Context)R0-r14, SPSR&pc from the stack of the restored task.
- Store the User Mode Stack Pointer (the new Task Stack Pointer of the interrupted task) to the TCB of the interrupted task (pointed to by User Mode Stack Pointer).
- Switch the processor to User Mode.
- @

6 ПРИПИНЕННЯ ЗАДАЧ

Agenda:

- 1) схеми потоків даних;
- 2) специфікації процесів.

6.1 Схеми потоків даних

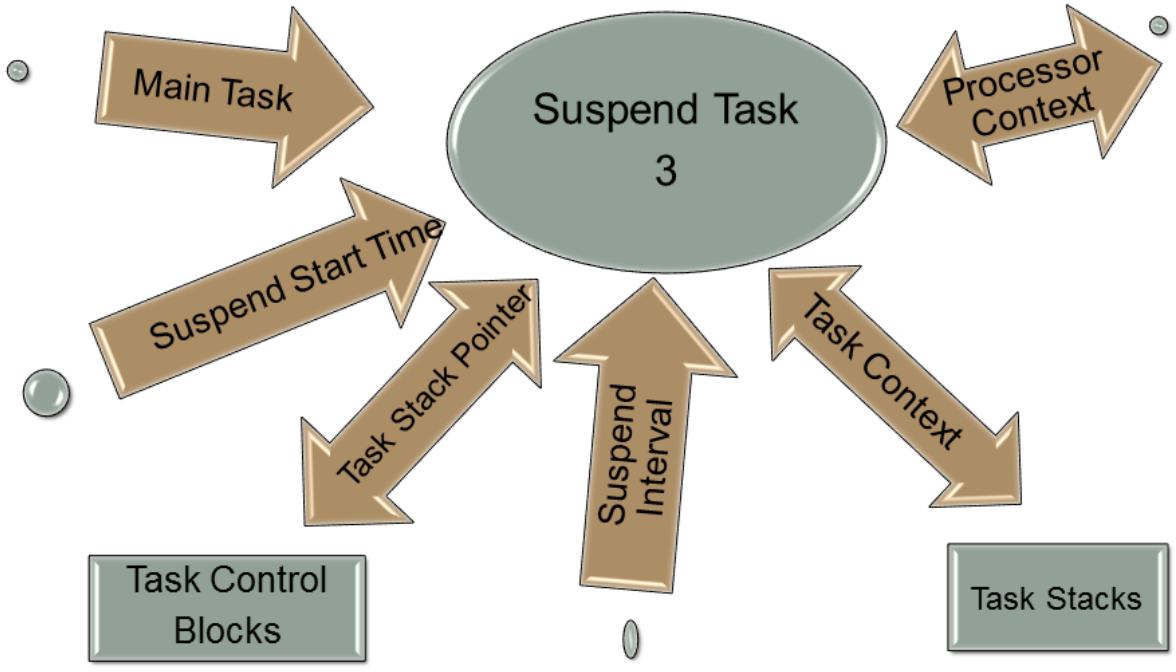
InformatiCup

www.informaticup.de

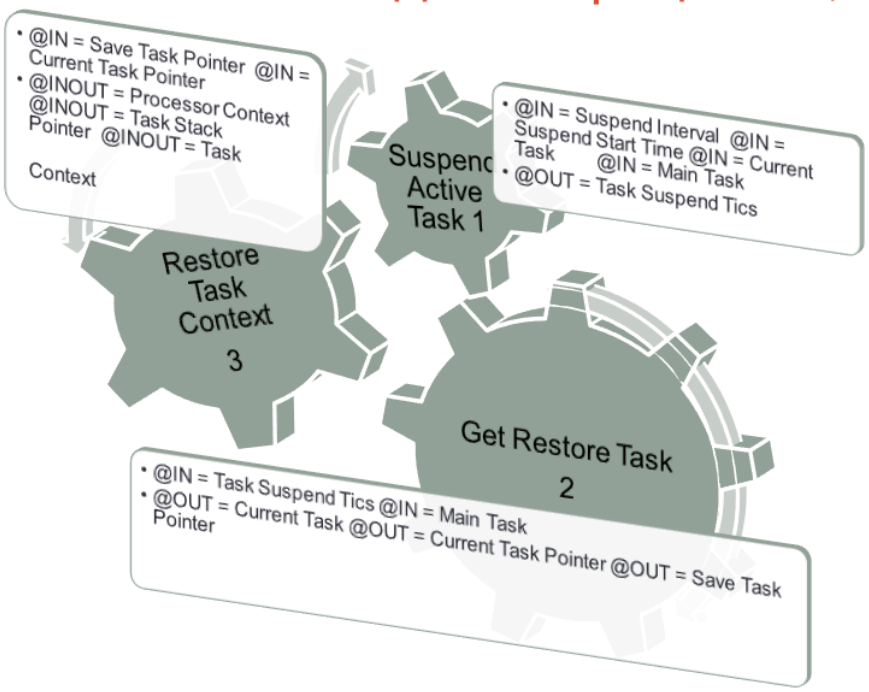
6.1 Припинення задач

- Задача може тимчасово зупинити себе в любу мить, але звичайно зупинка виконується коли завершено виконання всіх операцій для поточної її активації.
- Після зупинки задача з найвищім пріоритетом отримує управління мікропроцесору.

6.1 Схема потоків даних Suspend Task (Parent: GPS Architect OS)



6.1 Схема потоків даних процесів 1,2,3



6.2 Специфікації процесів

6.2 Специфікація процесу припинення активної задачі

- @IN = Suspend Interval @IN = Suspend Start Time
  @IN = Current Task @IN = Main Task
- @OUT = Task Suspend Tics
- @PSPEC Suspend Active Task
- //Can't suspend MAIN
- IF Current Task == Main Task
- RETURN
- Get current time.
- IF (current time – Suspend Start Time) < (Suspend Interval + 1)
- Task Suspend Tics for Current Task =
- Suspend Interval - (current time -Suspend Start Time)
- ELSE
- Task Suspend Tics for Current Task = 1
- ENDIF
- @

6.2 Специфікація процесу визначення активної задачі

- @IN = Task Suspend Tics @IN = Main Task
- @OUT = Current Task @OUT = Current Task Pointer @OUT = Save Task Pointer
- @PSPEC Get Restore Task
- Save Task Pointer = TCB Pointer of Current Task
- FOR Task = 0 TO MainTask)
- IF Task Suspend Tics for Task ==0)
- Current Task = Task
- break;
- ENDIF
- ENDFOR
- Current Task Pointer = TCB Pointer of Current Task
- @

6.2 Специфікація процесу збереження та поновлення програмної задачі

- @IN = Save Task Pointer @IN = Current Task Pointer
- @INOUT = Processor Context
  @INOUT = Task Stack Pointer @INOUT = Task Context
- @PSPEC Save And Restore Tasks
- Store the Processor Context (Task Context of the save task):
- R0-r14, SPSR&pc
- to the stack pointer of the TCB Save Task Pointer.
- Store the new stack pointer to the TCB Task Stack Pointer of the save task.
- Restore the processor context(Task Context of the restore task):
- R0-r14, SPSR&pc
- from the stack of the TCB pointed at by Current Task Pointer.
- @

6.3 Коментар до програми 1

```
/*************************************************************
* Function: void Suspend(unsigned SuspTics, unsigned
long StartTic)
* Revision: 6.02sa
* Last Update: 9/9/2011
* Suspends the current task for the given number if TICs.
* Input: SuspTics - the number of TICs for the suspension.
*        StartTic - the TIC count when the task started.
* Output: None.
* Return Value: None.
*************************************************************/
```


6.3 Програма 1а

```
void Suspend(unsigned SuspTics, unsigned long StartTic)
{
   int task_being_suspended;
   disable();
   /* MAIN task can't be suspended so just re-enable
interrupts and return. */
   if(CurrentTask==MainTask)  {enable(); return; }
   if((unsigned)(TIC - StartTic) < (SuspTics + 1))
   TCB[CurrentTask].TSuspTics = SuspTics -
  (unsigned)(TIC - StartTic);
   else   TCB[CurrentTask].TSuspTics = 1;
   task_being_suspended = CurrentTask;
```

6.3 Програма 1b

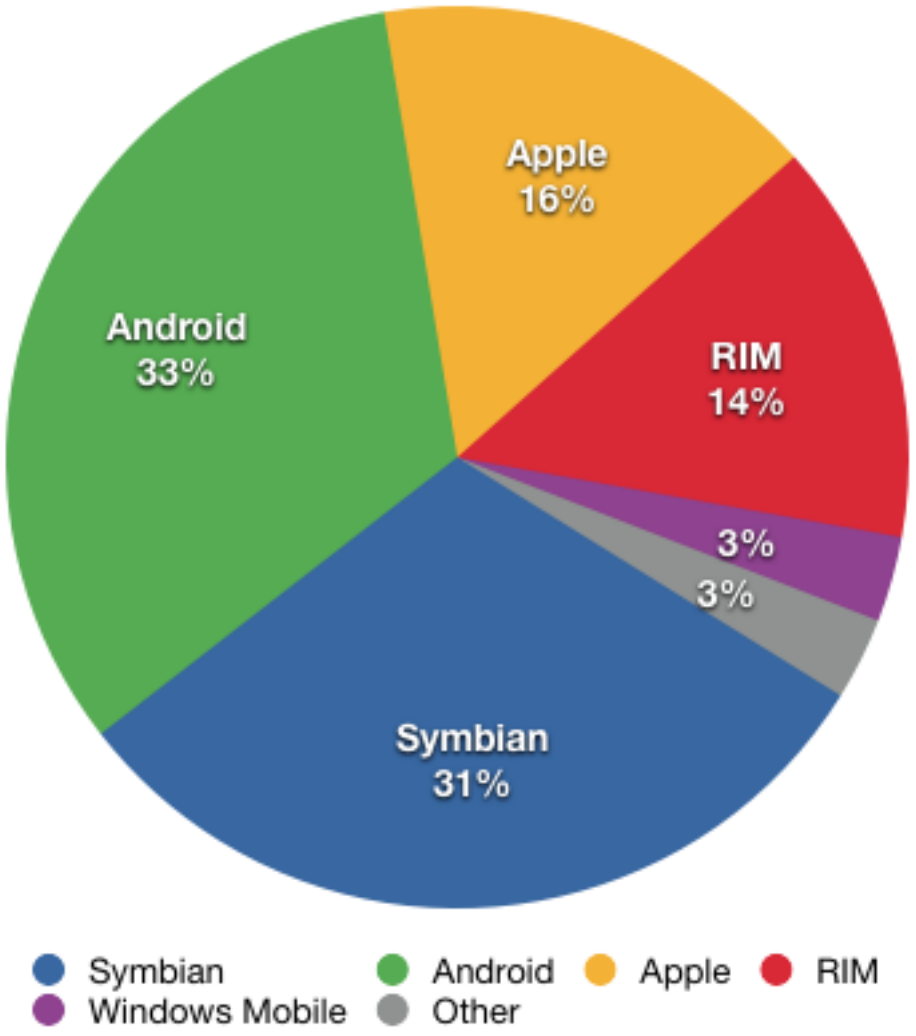
```
/* The new current task will be the highest priority task which has
a zero suspension interval.*/
    for( CurrentTask = 0;CurrentTask<=  MainTask; CurrentTask
  ++)
    {
        if(TCB[CurrentTask].TSuspTics==0)
            break;
    }
    CurrentTaskPointer = &TCB[CurrentTask];
/* Save the context of the task being suspended and restore the
context  of the new current task. */
  SaveAndRestore(&TCB[task_being_suspended],&TCB[Current
  Task]);
}
```

Тематичний модуль 3.
ТИПОВИЙ ПРИЙМАЧ

Лекція 7. Комплект для розробки приймача GPS

Лекція 8. Програмне забезпечення

Лекція 9. Сучасні тенденції



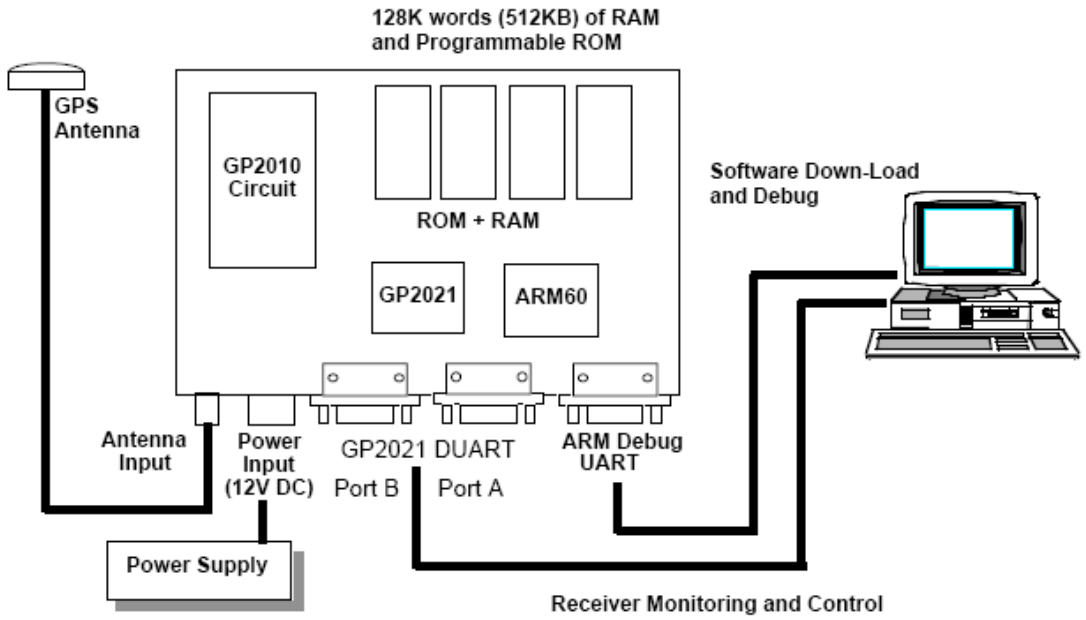
7 ЗАГАЛЬНА ХАРАКТЕРИСТИКА GPS

Agenda

- загальна характеристика комплекту GPS ARCHITECT;
- комплект мікросхем GP2000.

7.1 Загальна характеристика комплекту GPS ARCHITECT

Part No	Description	Data Sheet Reference
DW9255	35.42 MHz SAW Filter	DS3861
GP2010	GPS RF Front End	DS4056
GP2015	GPS RF Front End	DS4374
GP2021	12 Channel Correlator	DS4057
P60ARM-B	32-bit RISC Processor	DS3553



WINMON

COM Port Display Command About Help

NAVIGATION DATA

Lat	N	51°34.7838'	Spd	0.19	GDOP	1.4	SVs	11	HE	2.5	Date 04/02/97
Lon	W	1°46.1167'	Hdg	348.3°	PDOP	1.3	3D Nav		VE	-2.1	UTC 15:45:10
Hgt		161.97	ROC	-0.01	VDOP	1.0	DGPS		DO	149.1	OscErr -0.09

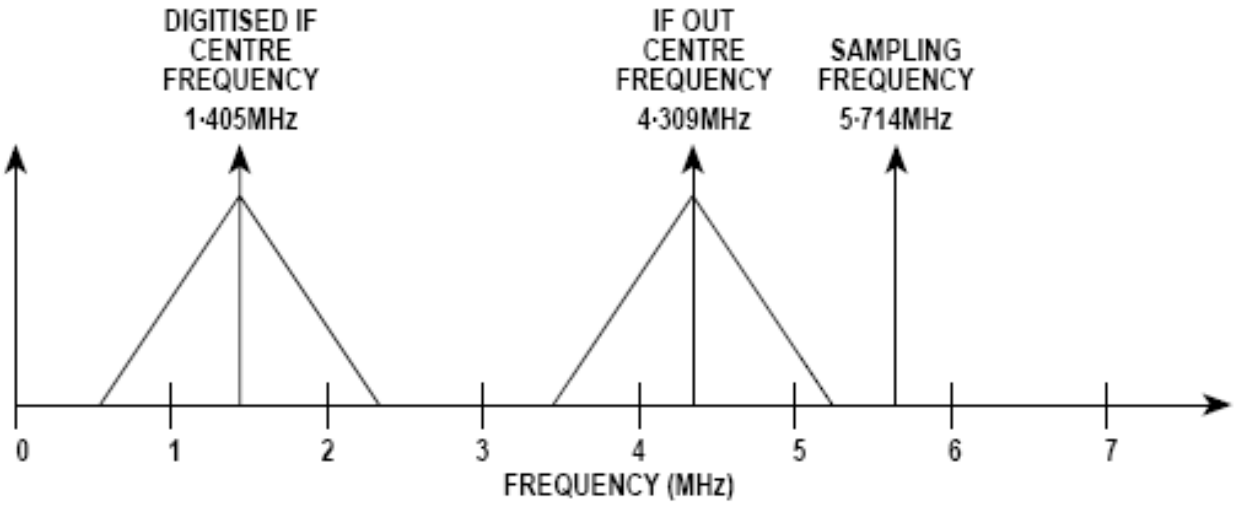
CHANNEL STATUS

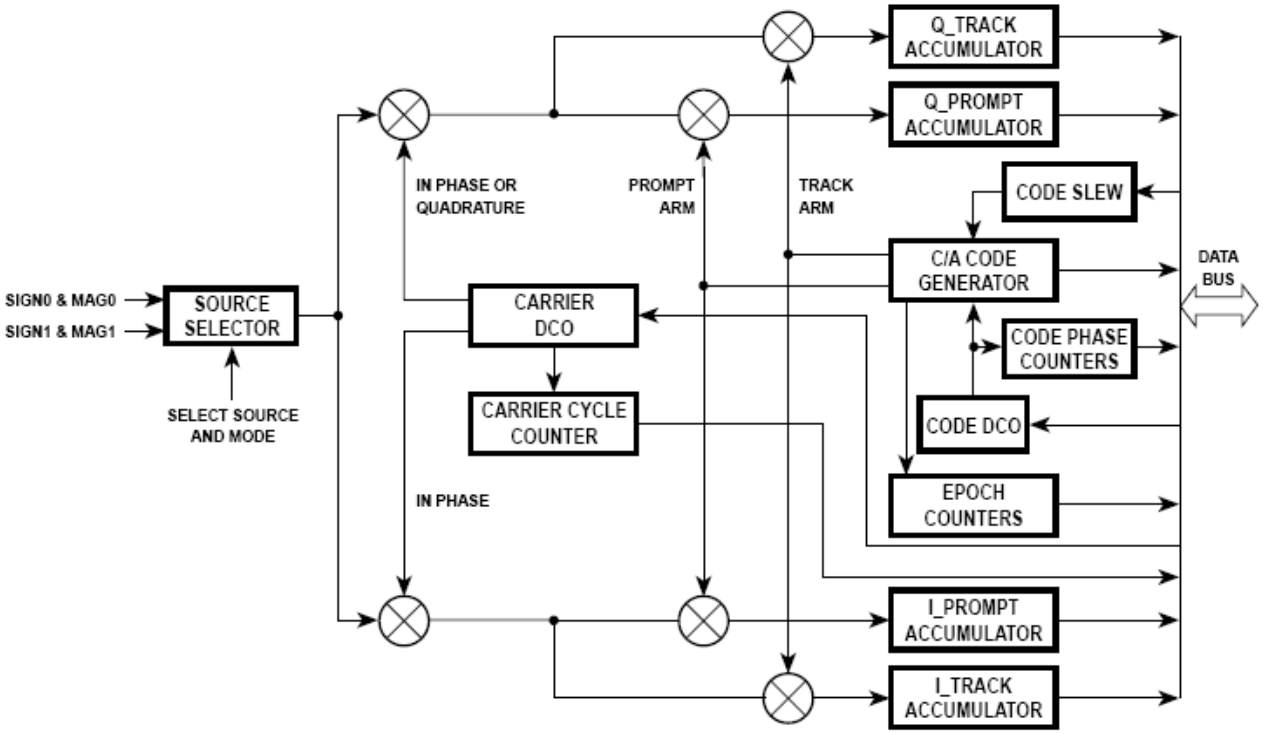
CH	SV	ELV	AZI	DOPP	NCO	UERE	SF	PRerr	PRRerr	ICPerr	DiffC	LOCK	SNR
1	1	25	226	-2928	-2780	32	3	0.1	0.0	0.0	29.6	CCBF	16.2
2	4	9	26	-2479	-2334	32	3	-3.9	0.2	0.0	-42.8	CCBF	11.1
3	25	48	295	2106	2254	32	3	-4.2	-0.1	0.0	-7.9	CCBF	17.1
4	6	64	187	1568	1714	32	3	-0.6	-0.1	0.0	63.6	CCBF	20.2
5	5	36	88	-2680	-2532	32	3	-0.4	0.0	0.0	5.4	CCBF	15.9
6	24	29	60	-385	-236	32	3	2.7	0.0	0.0	35.3	CCBF	16.8
7	29	14	324	1887	2035	32	3	-0.2	0.2	0.0	-21.3	CCBF	12.1
8	14	0	323	-1585	-1437	32	5	0.3	0.3	0.0	-10.0	CCBF	6.1
9	9	2	139	-3702	-3555	32	3	-3.6	0.1	0.0	-23.0	CCBF	7.0
10	16	2	350	-317	-162	32	1	-2.9	-1.1	0.0	-46.2	CCBF	6.5
11	30	70	77	-1105	-958	32	3	-3.5	0.0	0.0	-0.8	CCBF	18.7
12	22	0	272	3428	3576	0	0	0.0	0.0	0.0	0.0		3.0

SYSTEM STATUS

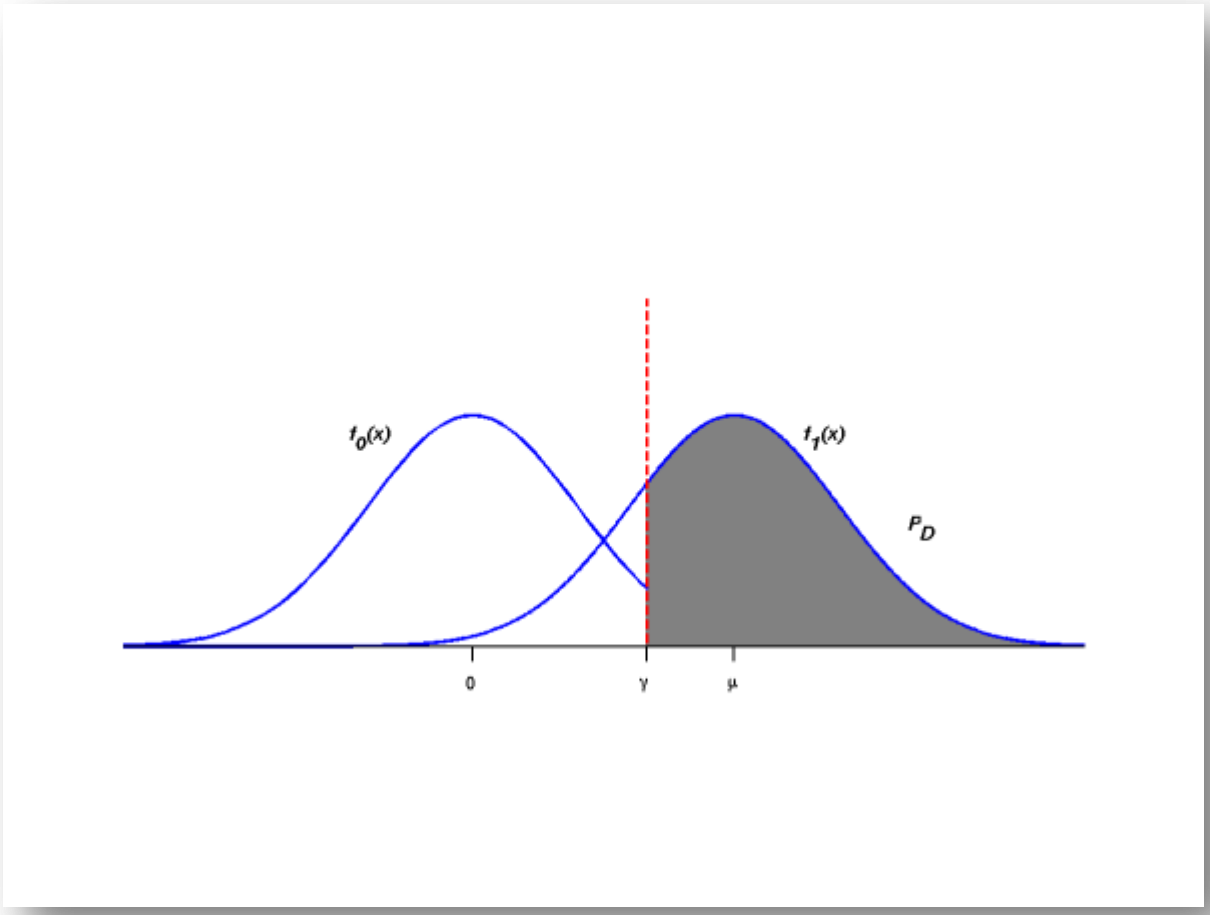
3-D NAVIGATION IN PROGRESS (11 PSEUDO-RANGES)

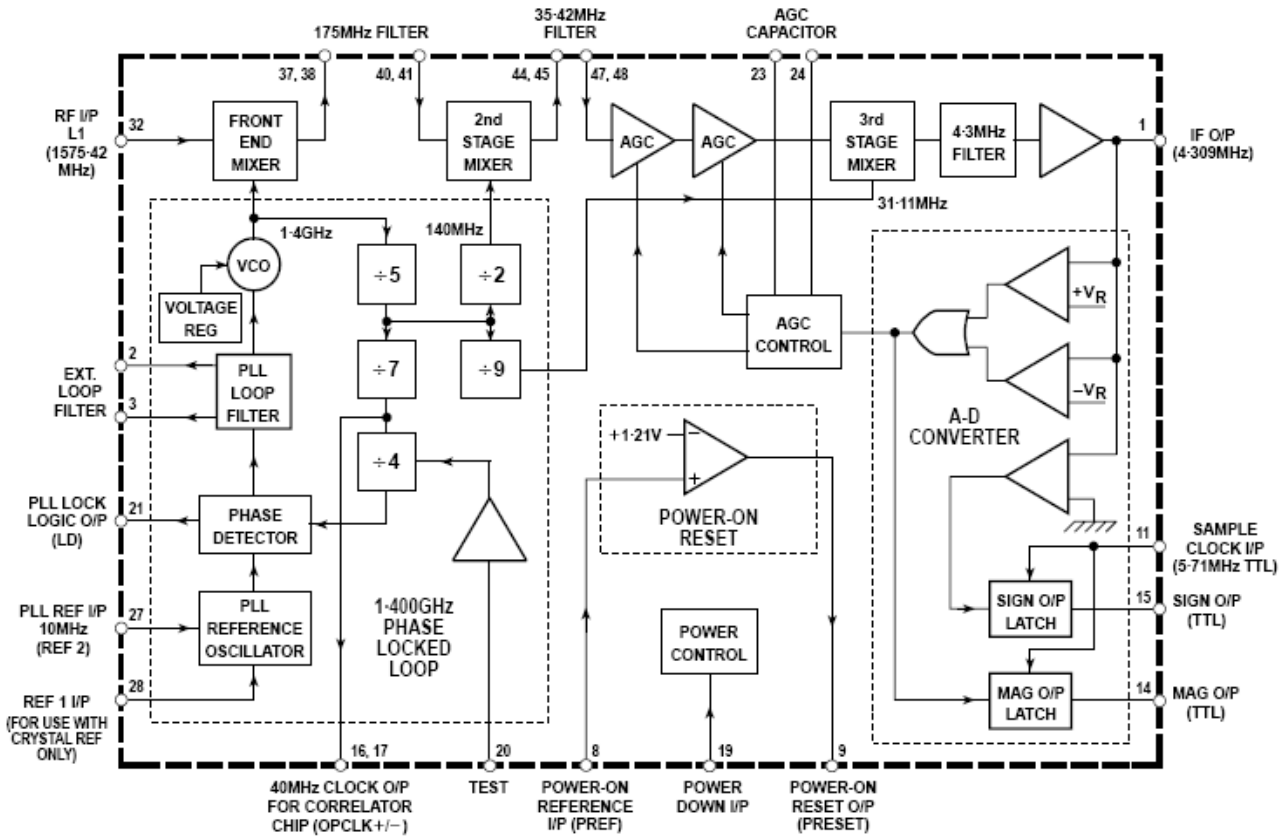
7.2 Комплект мікросхем GP2000

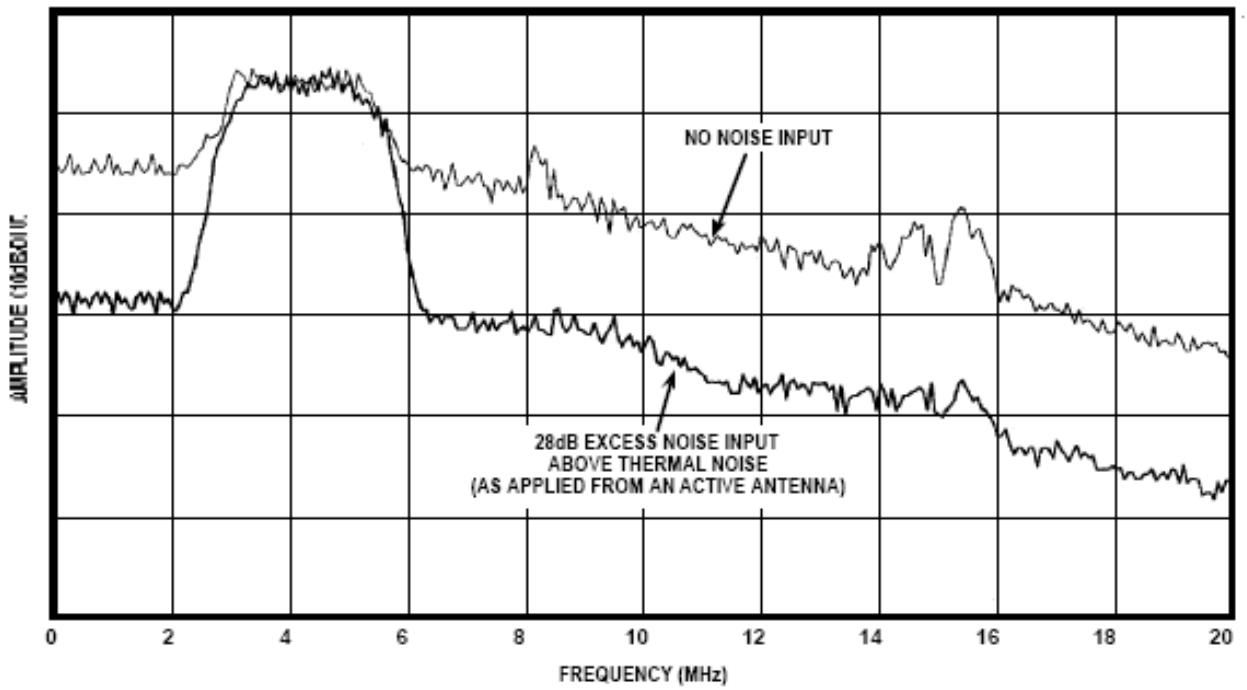


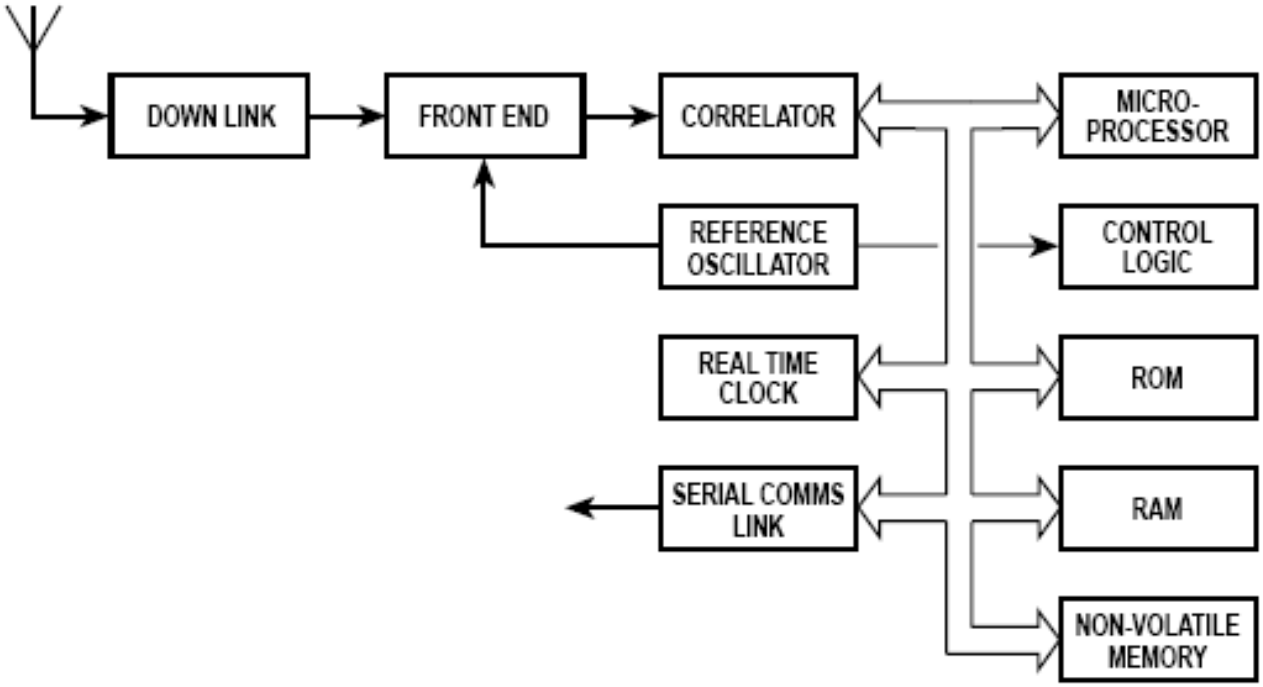


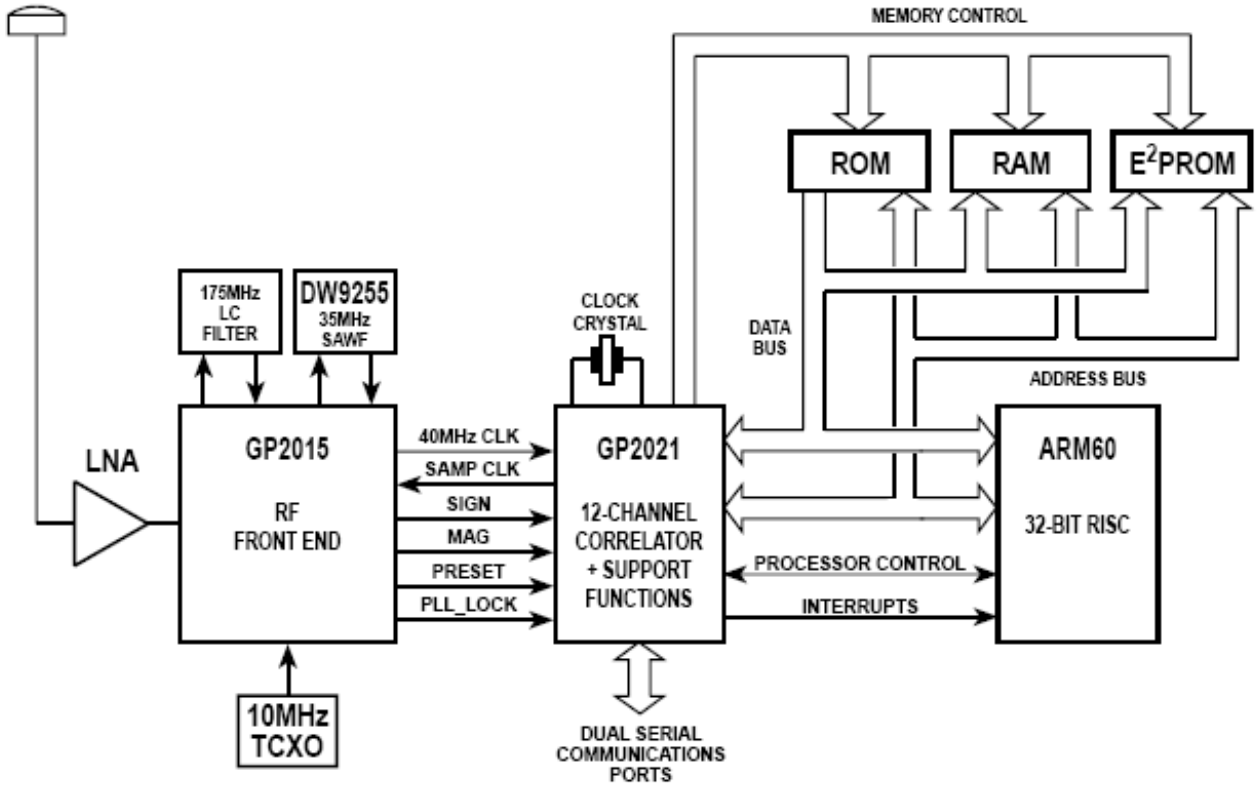
$$\Lambda(\mathbf{x}) = \frac{f_1(\mathbf{x})}{f_0(\mathbf{x})} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \eta$$

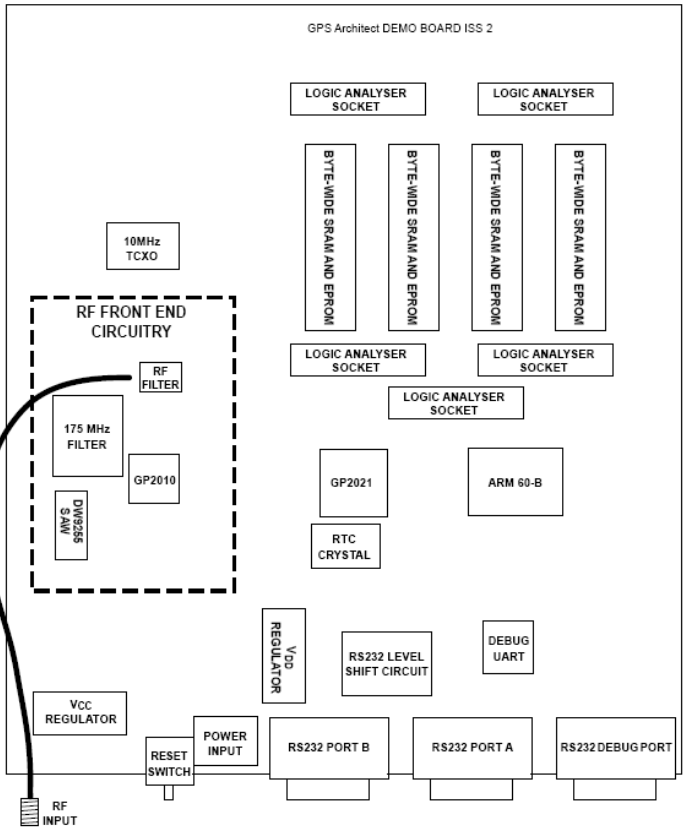

$f_0(x)$
$f_1(x)$
P_D
0
γ
μ


175MHz FILTER
35·42MHz FILTER
AGC CAPACITOR
37, 38
40, 41
44, 45
47, 48
23
24
RF I/P L1 (1575·42 MHz)
32
FRONT END MIXER
2nd STAGE MIXER
AGC
AGC
3rd STAGE MIXER
4·3MHz FILTER
1
IF O/P (4·309MHz)
31·11MHz
1·4GHz
140MHz
VCO
VOLTAGE REG
÷5
÷2
÷7
÷9
÷4
AGC CONTROL
+VR
–VR
A-D CONVERTER
2
EXT. LOOP FILTER
3
PLL LOOP FILTER
+1·21V
POWER-ON RESET
PLL LOCK LOGIC O/P (LD)
21
PHASE DETECTOR
11
SAMPLE CLOCK I/P (5·71MHz TTL)
SIGN O/P LATCH
15
SIGN O/P (TTL)
PLL REF I/P 10MHz (REF 2)
27
PLL REFERENCE OSCILLATOR
1·400GHz PHASE LOCKED LOOP
POWER CONTROL
MAG O/P LATCH
14
MAG O/P (TTL)
REF 1 I/P (FOR USE WITH CRYSTAL REF ONLY)
28
16, 17
20
8
19
9
40MHz CLOCK O/P FOR CORRELATOR CHIP (OPCLK+/–)
TEST
POWER-ON REFERENCE I/P (PREF)
POWER DOWN I/P
POWER-ON RESET O/P (PRESET)


NO NOISE INPUT
28dB EXCESS NOISE INPUT
ABOVE THERMAL NOISE
(AS APPLIED FROM AN ACTIVE ANTENNA)
0
2
4
6
8
10
12
14
16
18
20
FREQUENCY (MHz)


DOWN LINK
FRONT END
CORRELATOR
MICRO-
PROCESSOR
REFERENCE
OSCILLATOR
CONTROL
LOGIC
REAL TIME
CLOCK
ROM
SERIAL COMMS
LINK
RAM
NON-VOLATILE
MEMORY


MEMORY CONTROL
ROM
RAM
E2PROM
175MHz LC FILTER
DW9255 35MHz SAWF
CLOCK CRYSTAL
DATA BUS
ADDRESS BUS
LNA
GP2015
RF FRONT END
40MHz CLK
SAMP CLK
SIGN
MAG
PRESET
PLL_LOCK
GP2021
12-CHANNEL CORRELATOR + SUPPORT FUNCTIONS
ARM60
32-BIT RISC
PROCESSOR CONTROL
INTERRUPTS
10MHz TCXO
DUAL SERIAL COMMUNICATIONS PORTS


GPS Architect DEMO BOARD ISS 2
LOGIC ANALYSER SOCKET
LOGIC ANALYSER SOCKET
BYTE-WIDE SRAM AND EPROM
BYTE-WIDE SRAM AND EPROM
BYTE-WIDE SRAM AND EPROM
BYTE-WIDE SRAM AND EPROM
10MHz TCXO
RF FRONT END CIRCUITRY
RF FILTER
175 MHz FILTER
GP2010
DW9255 SAW
LOGIC ANALYSER SOCKET
LOGIC ANALYSER SOCKET
LOGIC ANALYSER SOCKET
GP2021
ARM 60-B
RTC CRYSTAL
VDD REGULATOR
RS232 LEVEL SHIFT CIRCUIT
DEBUG UART
VCC REGULATOR
RESET SWITCH
POWER INPUT
RS232 PORT B
RS232 PORT A
RS232 DEBUG PORT
RF INPUT

8 ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

Agenda:

- Загальна характеристика
- Алгоритм обробки сигналу

8.1 Загальна характеристика

Програмне забезпечення GPS ARCH формує бінарне зображення(EXE файл), яке має бути завантажено з комп'ютера PC до середовища ARM Toolkit.

Пакет ARM Toolkit використовує мову програмування C для розробки вихідного коду.

7.1 Атрибути програмного забезпечення

340 KBytes of ‘C’ source code

82 KBytes of ‘H’ include files

8 KBytes of ‘S’ assembler files

157 KBytes executable (40K332bit)

Compiles with ARM Toolkit v2.0

Конкуруючі завдання.

Програмне забезпечення виконує наступні конкуруючі завдання:

Прочитати акумулятори каналу корелятора в GP2021

Управляти каналом корелятора, що відстежує петлі

Контролювати умови замикання петлі

Оновлювати оцінки позиції та швидкості приймача

Аналізувати дані GPS, одержані від супутників

Збирати блоки вимірювання

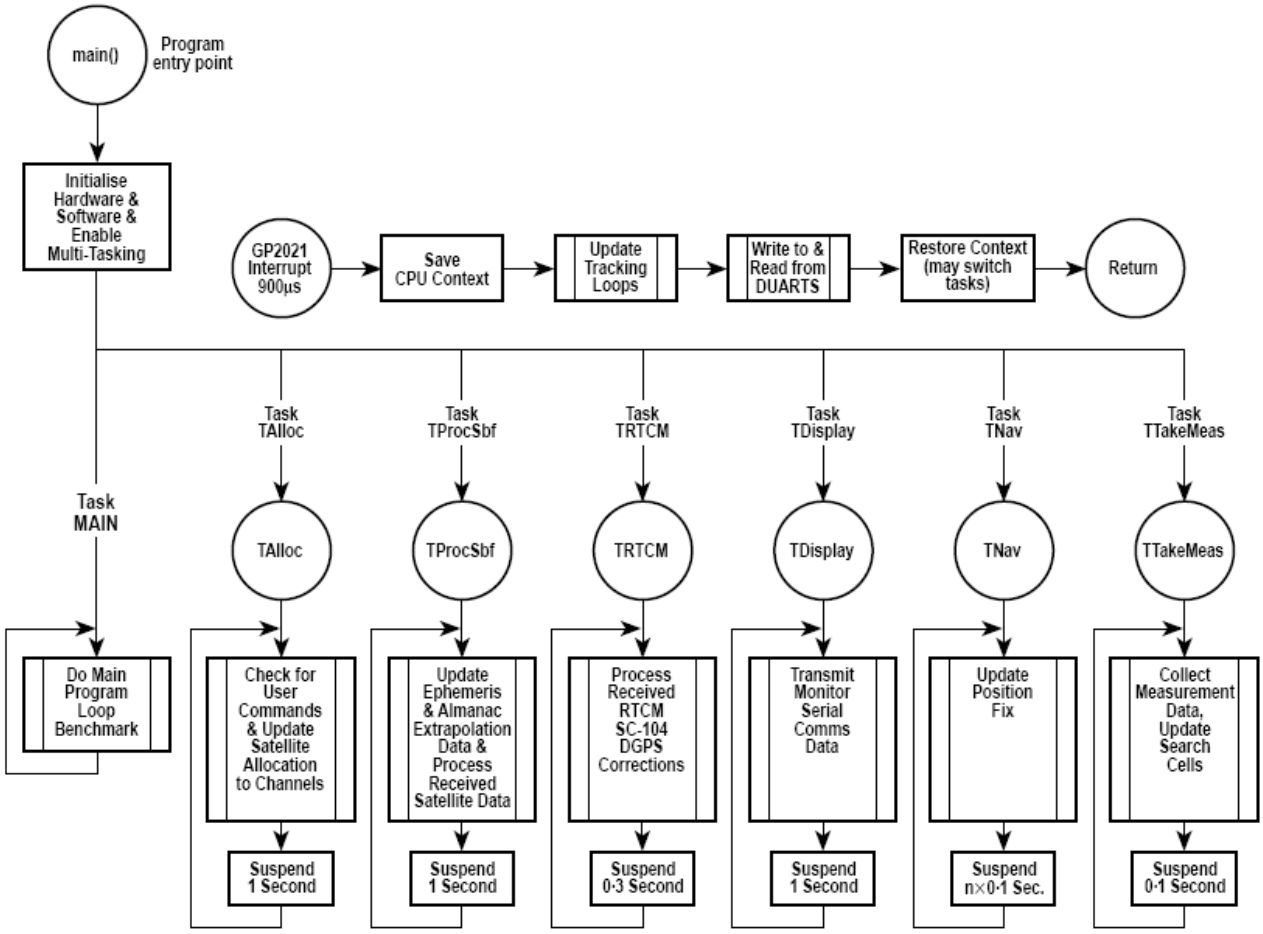
Розраховувати позицію супутника щодо спостерігача, засновану на ефемеридних даних

Управляти стратегією вибору супутника

Формувати потік вихідних даних RS232

Перевіряти існування даних RTCM SC-104 DGPS для виправлення і аналізу даних.

Операційна система перемикання завдань.



Програмне забезпечення діє як просте перемикання завдань операційною системою, щоб забезпечити дію конкуруючих структур, які надають собі час щодо вимог обробки сигналу GPS і програмне забезпечення.

Операційна система перемикання завдань складається з переривань і процедур обробки завдань. Тільки одне завдання може бути активне в будь-який час.

Головне джерело – це переривання одержане від GP2021, сигнал ACCUM_INT (типове значення періоду 505,05µs - у програмі GPS ARCH встановлюється значення 900,025µs) обслуговується програмою обробки переривання (Interrupt Service Routine-ISR). Процедура ISR управляє обслуговуванням операційної системи GPS ARCH і обробкою даних сирих накопичень GP2021 для підтримки сигналу корелятора, що відстежує петлі.

Управління завданнями.

Архітектор GPS має 7 певних завдань і 1 рутина обслуговування переривання. Нові завдання можуть легко бути додані споживачем.

Завдання можуть бути або активними, або зупиненими. Тільки одне

завдання обслуговується мікропроцесором в один момент.

Кожне конкуруюче завдання програмного забезпечення бере одну із загальних двох форм:

1). Завдання, яке винне бути відновленим у фіксовані інтервали часу

2). Завдання, яке винне бути підвішеним у фіксовані інтервали часу

Пріоритет завдання визначає яке завдання одержує процесор протягом конфлікту завдань. Завдання Main()завжди визначене, щоб мати найнижчий пріоритет. Затримання процесора може бути досягнуте, виводячи з ладу перемикання задач або блокуючи (маскуючи) переривання GP2021. Рутина припинення завдання діє в межах операційної системи щоб тимчасово зупинити дію завдання для певної кількості TlCs (1TlC – 0,0999999s).

Масив TCB.

Завдання можуть бути додані або видалені з програмного забезпечення, виправляючи глобальну структуру Блоку управління завданнями (Task Control Block-TCB). Порядок завдань в декларації TCB є пріоритетом завдання.

Для кожного завдання, TCB має наступні записи:

- 1) покажчик на адресу початку завдання;
- 2) покажчик на межу стеку (ініціалізація NULL);
- 3) покажчик вершини стека (ініціалізація 0);
- 4) ім'я завдання;
- 5) розмір стека завдання;
- 6) інтервал зупинки завдання.

8.2 Алгоритм обробки сигналу

Знаходження коду і стеження за ним.

Для знаходження коду, обидва і фаза коду GP2021 і частота носія повинні відповідати фазі коду і частоті носія прийнятого сигналу на такій тривалості, щоб кореляція була вище порогу виявлення.

GPS ARCH виконує пошук у площині фаза коду, несуча частота, перебираючи всі кодові фази в серіях частотних діапазонів, поки сигнал не виявлений.

Як тільки сигнальне виявлення сигналу відбувається, цикли пошуку коду і носія закриваються.

Для стеження за кодом використовується Фазова Замкнута Петля.

Для стеження за носієм використовується Частотна Замкнута Петля.

Петля, що відстежує носій, допомагає петлі, яка відстежує код.

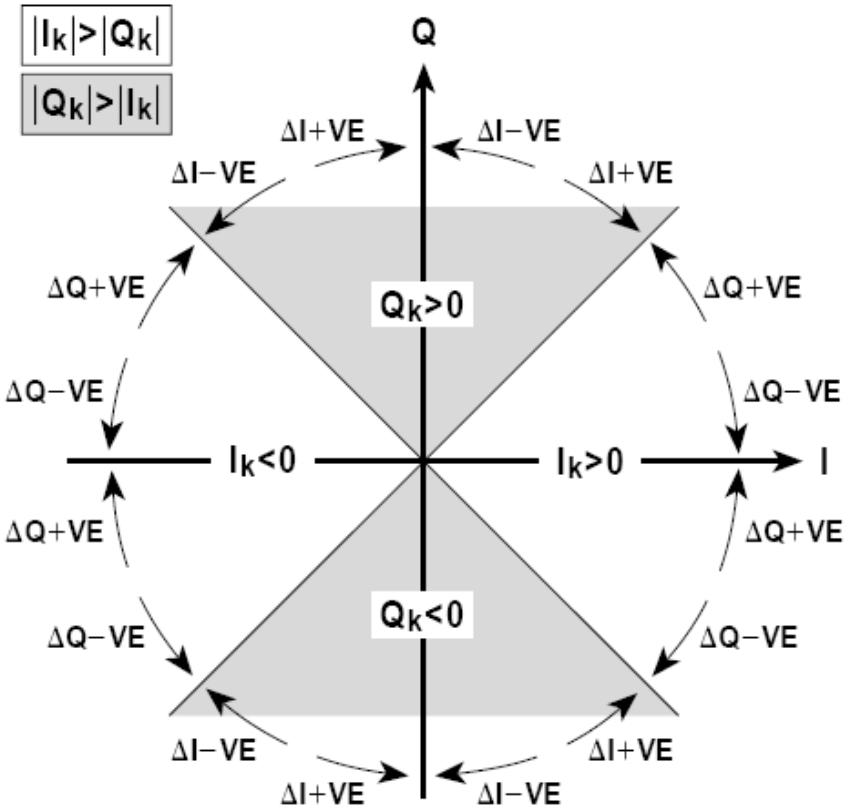
Пошук коду.

GPS ARCH використовує точну копія, яка ковзає для пошуку кодового придбання.

Частота GP2021 генератору коду DCO програмується злегка вище частоти формування чипів, яка передбачена таким чином, що коди з часом ковзають один за одним. Типове програмоване значення надає пошукову швидкість 0,25 чипа за мілісекунду.

Відтепер, повний пошук коду (1023 чипи) у наданій частоті займає близько 4 секунд для завершення. Частотні діапазони мають 500Hz завширшки.

Тому, щоб знайти частоту у просторі +-10,25kHz необхідно близько 164 секунд для завершення.



9 СУЧАСНІ ТЕНДЕНЦІЇ

Agenda

- короткий огляд процедури обслуговування переривання;
- короткий огляд задачі TTakeMeas.

9.1 Короткий огляд процедури обслуговування переривання

Процедура[7] ISR виконує два основних набора функцій: обслуговування операційної системи та обробка даних, які накоплені в мікросхемі GP2021, з метою відтворення прийнятого інформаційного сигналу. Далі розглядається тільки обробка даних, які накопичені в мікросхемі GP2021

Процедура ISR активується перериванням ACCUM_INT, яке надходить з мікросхеми GP2021 з періодом 900.025μs, та має самий високий пріоритет у порівнянні з любою прикладною задачею програмного забезпечення.

Процедура ISR повинна бути виконана до повторного виконання за перериванням ACCUM_INT(non-re-entrant). Очевидно, що час на виконання рутини ISR повинен бути значно менше періоду формування переривань ACCUM_INT, щоб залишився час на виконання інших прикладних програмних задач.

Функції ISR.

Програма GPS ARCH збирає данні I,Q з тих каналів,які доступні для накопичення нових даних. Ці данні використовуються в ряді функцій:

1) визначення значень лічильників епох коду та несучої;

2) визначення індикаторів блокування коду та несучої;

3) стеження за несучою;

4) стеження за кодом;

5) збереження бітових даних повідомлень від супутників.

Процедура ISR активізується ACCUM_INT, тому вона має самий високий пріоритет над усіма програмними задачами.

Тому немає необхідності підтримувати методологію активізації і зупинки задач, які застосовуються іншими задачами програмного забезпечення GPS Architect.

При кожній активації ISR, перша операція – це збір накопичених даних від активних каналів, в яких доступні нові накопичені дані.

[7] Interrupt Service Routine – ISR

Поява нових накопичених даних в каналах асинхронне один до одного та до переривання ACCUM_INT, тому необхідно виконувати опит активних каналів на наявність нових даних.

Регістр стану GP2021 ACCUM_STATUS_A має 12-розрядну карту біт каналів, які мають нові накопичені данні.

Приймання I,Q

При кожній активації ISR, перша операція – це збір накопичених даних від активних каналів, в яких доступні нові накопичені дані.

Поява нових накопичених даних в каналах асинхронне один до одного та до переривання ACCUM_INT, тому необхідно виконувати опит активних каналів на наявність нових даних.

Регістр стану GP2021 ACCUM_STATUS_A має 12-розрядну карту біт каналів, які мають нові накопичені данні.

Розрахунок часу передавання даних.

Важливо щоб читання і збереження нових накопичених даних для всіх каналів відбувалося якомога швидше після переривання ISR, щоб збільшити період повторення ACCUM_INT і зменшити навантаження мікропроцесора.

Номінальний час накопичення коду C/A – 1ms. Якщо час читання і збереження усіх 12 каналів складає ΔT, то період повторення ACCUM_INT необхідно встановити рівним (1-ΔT) ms.

Фактично період буде ще меншим з урахуванням часу входження до ISR і необхідну обробку перед читанням суматорів.

У кожному каналі необхідно прочитати 4 суматора (In-phase Prompt, In-phase Track, Quadrature Prompt, Quadrature Track), тому максимально буде прочитано 48 накопичень.

Склад специфікацій: приймання I,Q; приймання лічильника епохи; перевірка TIC; поновлення лічильника пропущених накопичень; стеження за індикаторами захвату; поновлення параметрів петлі стеження за кодом; поновлення параметрів петлі стеження за несучою; формування потоку даних.

9.2 Короткий огляд задачі TTakeMeas

Первинні функції задачі TTAKEMEAS:
1) керування процесом пошуку частотного інтервалу, в якому знаходиться несуча частота сигналу;
2) збір необроблених даних з каналу стеження.

Також підтримуються процеси швидкого входження в синхронізм, коли

доступна відповідна інформація.

Інтервал активації задачі – 1ТІС.

10 КОНТРОЛЬНІ ЗАВДАННЯ ТА ЗАПИТАННЯ

Скласти специфікацію процесу при виконані функції: Initialise Tasks ().

Скласти специфікацію процесу при виконані функції: Initialise Context ().

Скласти специфікацію процесу збереження контексту мікропроцесору при виконані функції: IRQ Handler()

Скласти специфікацію процесу зміни задачі при виконані функції: GPISR()

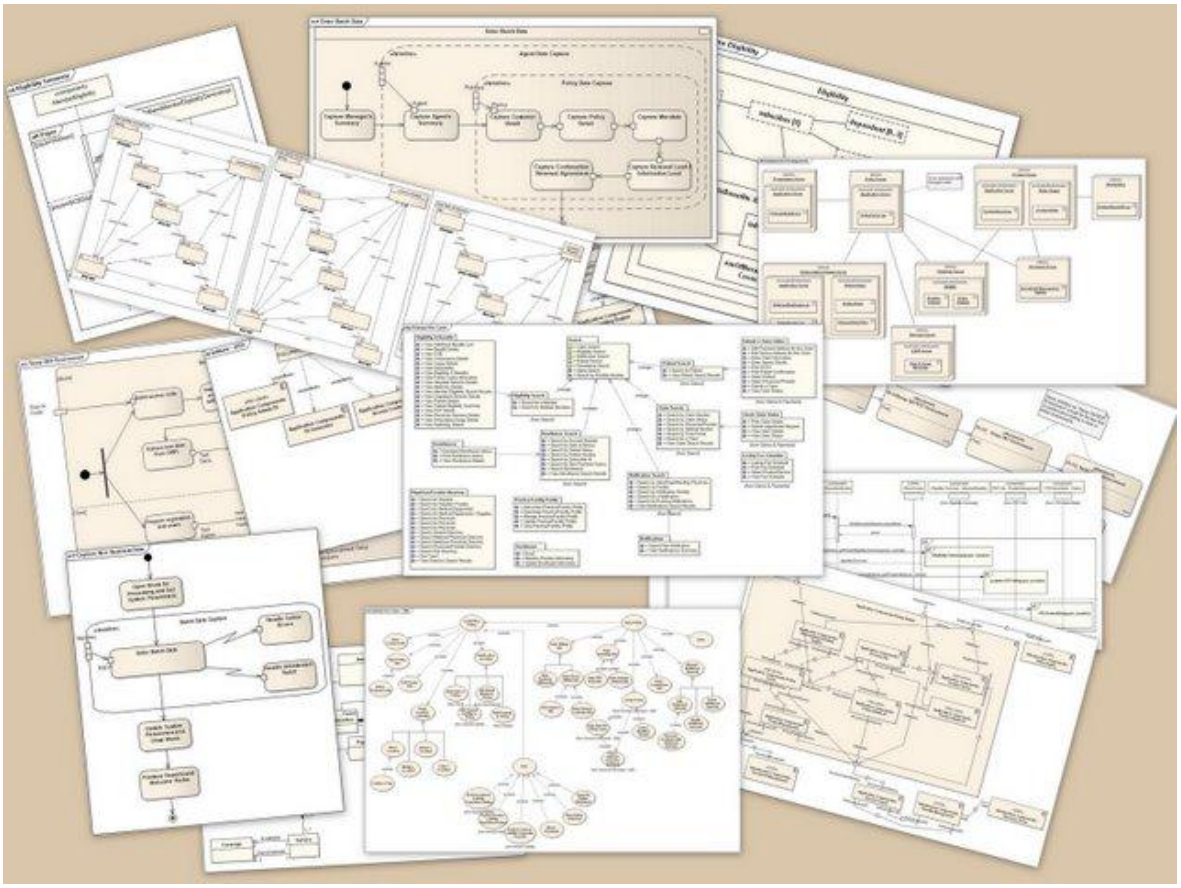
Скласти специфікацію процесу поновлення контексту мікропроцесору процесу при виконані функції: IRQ Handler ()

Скласти специфікацію процесу зупинки активної задачі при виконані функції: Suspend ()

Скласти специфікацію процесу визначення активної задачі при виконані функції: Suspend ()

Скласти специфікацію процесу збереження та поновлення контексту мікропроцесору при виконані функції: Save And Restore ().

Висновки



ПЕРЕЛІК ЛІТЕРАТУРИ

1. Яценков, В.С. Основы спутниковой навигации. Системы GPS NAVSTAR и ГЛОНАСС. – М.: Горячая линия – Телеком, 2005. – 272с.: ил.

2. Скляр, Б. Цифровая связь. Теоретические основы и практическое применение. Изд.2-е, испр.: Пер с англ./ Б. Скляр – М.: Издательский дом «Вильямс», 2007. – 1104с.: ил.

3. Сергиенко, Б.А. Цифровая обработка сигналов/ А.Б.Сергиенко − СПб.: Питер, 2007. – 608с.: ил.

4. Бондарев, В.Н. Цифровая обработка сигналов: методы и средства. Учебное пособие для вузов 2-изд./ В.Н. Бондарев, Г. Трестер, В.С. Чернега - Х.: Конус.2001. - 398с

5. Product information and application notes. Mitel Semiconductors. TECHNICAL DOCUMENTATION GP2000 GPS Chipset – Designer’s GuideSupersedes Issue 1.4 in August 1996 Global Positioning Products Handbook, HB3045-1.0 MS4395-2.3 April 1998. © Mitel Corporation 1998 Publication No. MS4393 Issue No. 2.3 April 1998. - 1 електрон. опт. диск (CD-ROM). - Загл. с етикетки диска