Actividad 6.1

1) El lenguaje de programación OCCAM es un lenguaje de programación concurrente y de comunicación (CSP) entre procesos ideado inicialmente para la programación del hardware de procesamiento con memoria compartida como los Transputers INMOS (Array de procesadores homogéneos). Esto permite generar procesamientos paralelos en un solo procesador por medio del envío de señales o mensajes entre procesos. En ejemplos encontrados, mensionan el ejemplo de un proceso Front_End (con el cual tiene interacción el usuario) y un Back_End (en el cual se llevan a cabo los procesos). Ambos se comunican por mensajes que se leen con "?" y se responden o escriben con "!".

A modo informativo, agrego extracto de la referencia
web: http://www.geocities.ws/mofern/PaperOccamJAIIO_s.htm
pdf: http://www.geocities.ws/mofern/PaperOccamJAIIO_s.pdf

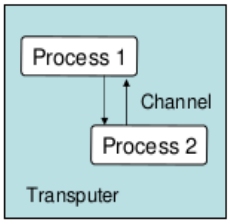
2 Generalidades del Lenguaje Occam

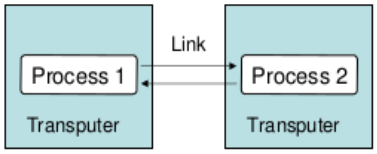
Occam es un lenguaje concurrente, basado en el CSP de Hoare. Está diseñado para soportar aplicaciones concurrentes en las cuales muchas partes de un sistema operan independientemente e interactúan. Un programa se expresa en términos de procesos concurrentes que se comunican enviando y recibiendo mensajes a través de canales de comunicación. Occam provee comunicación sincrónica entre procesos que se ejecutan en paralelo, selección no determinística y creación dinámica de procesos secuenciales. Además provee facilidades para programación en tiempo real, a través de constructores que permiten un manejo preciso del tiempo con prestaciones de timeout y alarmas. En los siguientes párrafos se describen las construcciones principales del lenguaje, recurriendo a ejemplos cuando sea necesario.

Procesos Primitivos: Estos procesos consisten en una acción elemental, y pueden combinarse para formar procesos arbitrariamente complejos. Los procesos primitivos consisten en asignación, entrada y salida de canales y bloqueo durante un período determinado de tiempo. El proceso `var := exp` asigna a la variable `var` el valor de la expresión `exp` y termina; los procesos `canal ? var` y `canal ! exp` leen y escriben respectivamente un valor por un canal de comunicación entre procesos, terminando solamente cuando la operación se haya completado (es decir que tanto la lectura como la escritura son *sincrónicas*). Como extensión al lenguaje Occam estándar, se definieron e implementaron extensiones que soportan entrada/salida con el usuario en un ambiente UNIX. Dichas extensiones toman la forma de dos canales predefinidos (`IN` y `OUT`) cuya función es comunicar a los procesos que los usen con el `stdin` y `stdout` de UNIX respectivamente. Por otra parte, el proceso `WAIT NOW AFTER exp` termina cuando el valor del reloj local sobrepasa el valor de la expresion `exp`. Además se provee un proceso `SKIP`

Los Transputers nacieron como nodos de procesamiento individuales con capacidad de comunicarse con otros 4 por medio de señales con el fin de procesar de manera paralela. Cada uno contaba con una pequeña memoria intertna y los mensajes funcionaban como intercambiadores de datos para continuar ciertos procesos en otro lado. Se planteaban redes de reloj comunes de 5MHz y podían trabajar internamente con hasta 80MHz por medio de PLLs propios.

Fueron allá por los 80 y pico los precursores de los MicroControladores con pequeñas memorias built-in para procesamiento intra-chip o intra-process (comunicados por *channels*). Los mensajes y señales *(links)* eran generados para relegar tareas inter-chip o inter-procesors.

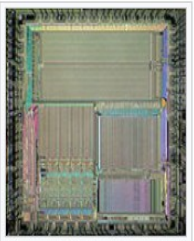




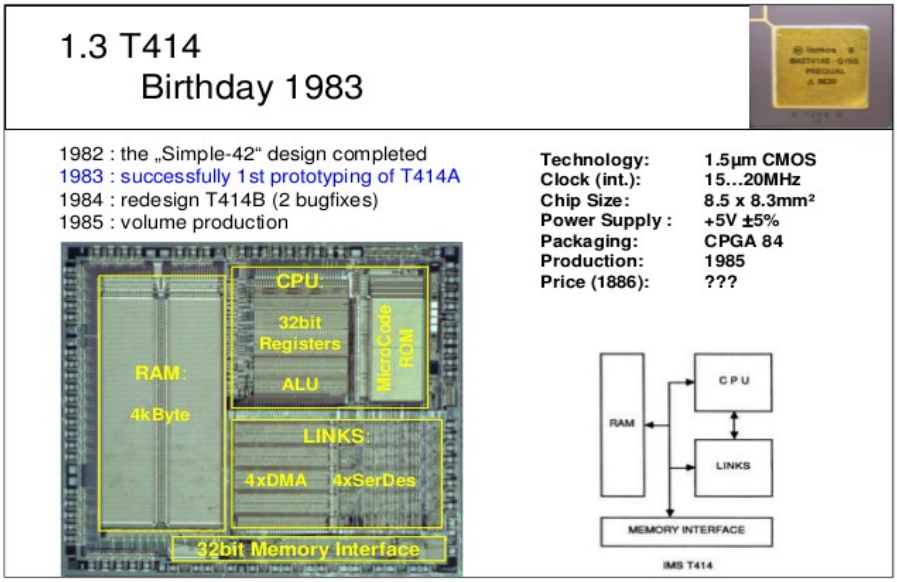
El Transputer T414 fue el primero de 32-bits con una memoria RAM interna de 2kB

T4: 32-bit [edit]

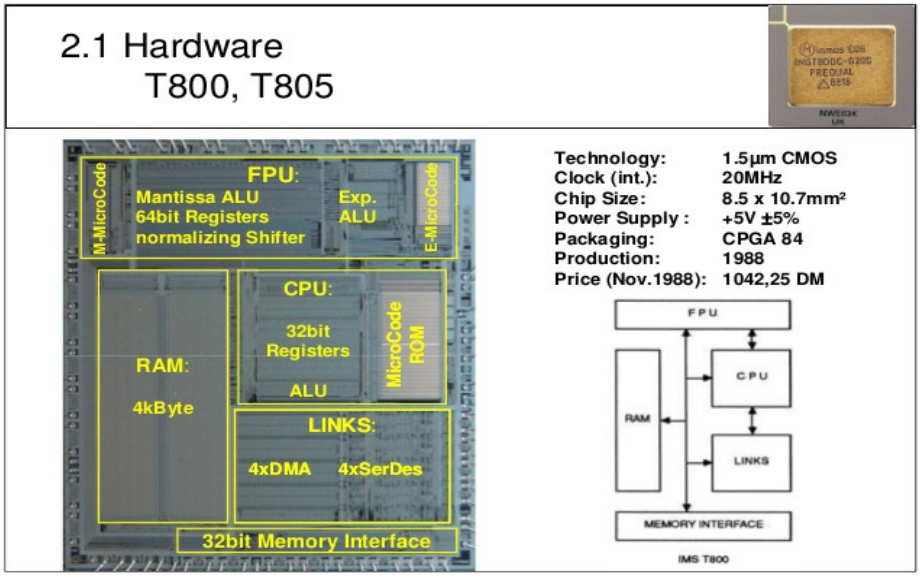
At launch, the **T414** was the 32-bit offering. Originally, the first 32-bit variant was to be the **T424**, but fabrication difficulties meant that this was redesigned as the T414 with 2 kB on-board RAM instead of the intended 4 kB. The T414 was available in 15 and 20 MHz varieties. The RAM was later reinstated to 4 kB on the **T425** (in 20, 25 and 30 MHz varieties), which also added the J 0 breakpoint support and extra T800 instructions. The **T400**, released in September 1989, was a low-cost 20 MHz T425 derivative with 2 kB and two instead of four links, intended for the embedded systems market.



Inmos T425 die



Dado su robustez, hoy en día se usan T805 por ejemplo en tecnología espacial.



2) En Transputers, la comunicación para comunicación entre procesos de hardware se realiza por medio de Inmos Protocol, o bien también por medio de RS232C. Para esto se utilizarían métodos y componentes ya conocidos y estudiados para la sincronización y recuperación de reloj y datos. Investigando encontré que ya se han desarrollado IPCores que simulan el funcionamiento de un T424 llamados TPCOREs.

3) Las FSM (Finite-State Machine) como las PN (Petri Net) son modelazaciones gráficas de máquinas de estados que a nivel funcionamiento presentan las mismas funcionalidades. La diferencia principal radica en la capacidad de verificación de estas técnicas. Las FSM basadas en PN permiten validar su funcionamiento cuando estas puedan presentar situaciones de concurrencia o conflicto, ya que en las PN solo una transición es posible, sin dar lugar a ejecución de procesos en paralelo que no deberían estarlo.

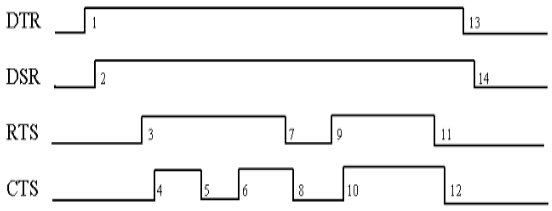
Investigando encontré mucha información sobre diagnosticablilidad sobre PN y observabilidad de estados (o lugares), que nunca había visto para el análisis de FSM puras.

Por último, también para profundizar estos análisis, existe CPN (Coloured Petri Net), el cual no solo analiza la movilidad de los tokens dentro de la red, sino que también, mediante colores, puede identificarlos y así analizar el origen e implicancia de cada uno de ellos. Un ejemplo de uso de esto es la verificación de diagnosticabilidad bajo el protocolo Token Ring.

4) Los lugares serán representados por componentes con memoria que puedan almacenar uno o más tokens. Algunos de estos pueden ser:

- Flip-Flops
- Counters
- Shift Registers
- RAM

5) Para hacer un ejemplo de Handshake implementé el Handshake de 4 fases de comunicación RS232, protocolo el cual utiliza las líneas RTS/CTS, DTR/DSR como semáforos del propio linkeo de los sistemas. Una descripción mínima se encuentra en lo imagen.



1 The computer sets DTR to indicate that it wants to make use of the modem.

2 The modem signals that it is ready and that a connection has been established.

3 The computer requests permission to send.

4 The modem informs the computer that it is now ready to receive data from the computer and send it through the phone wires.

5 The modem drops CTS to signal to the computer that its internal buffers are full; the computer stops sending characters to the modem.

6 The buffers of the modem have been purged, so the computer may continue to send data.

7 This situation is not clear; either the computer's buffers are full and it wants to inform the modem of this, or it doesn't have any more data to be send to the modem. Normally, modems are configured to stop any transmission between the computer and the modem when RTS is dropped.

8 The modem acknowledges RTS by dropping CTS.

9 RTS is again raised by the computer to re-establish data transmission.

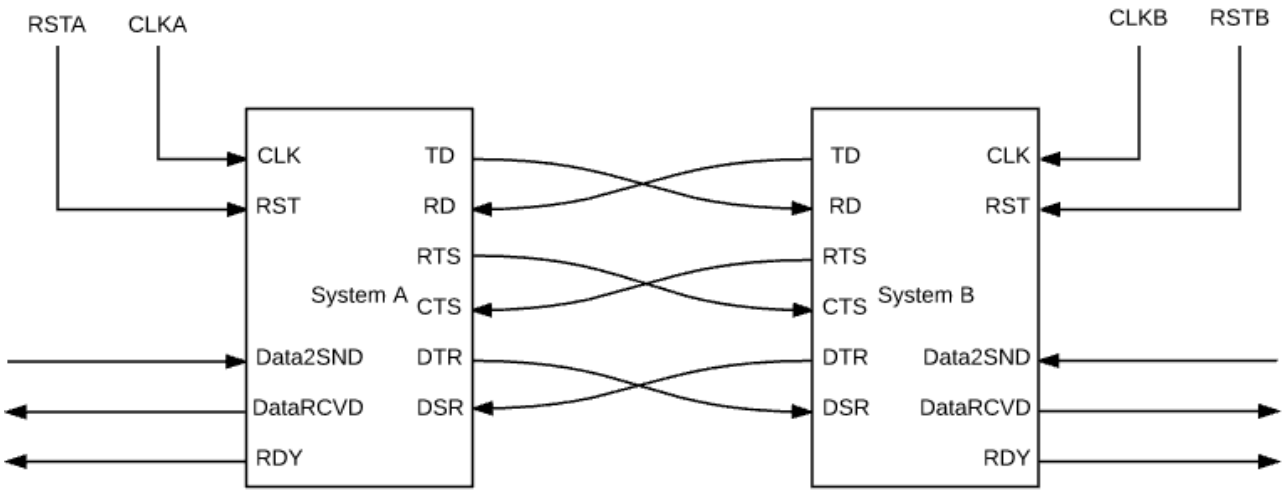
10 The modem shows that it is ready to do its job.

11 No more data is to be sent.

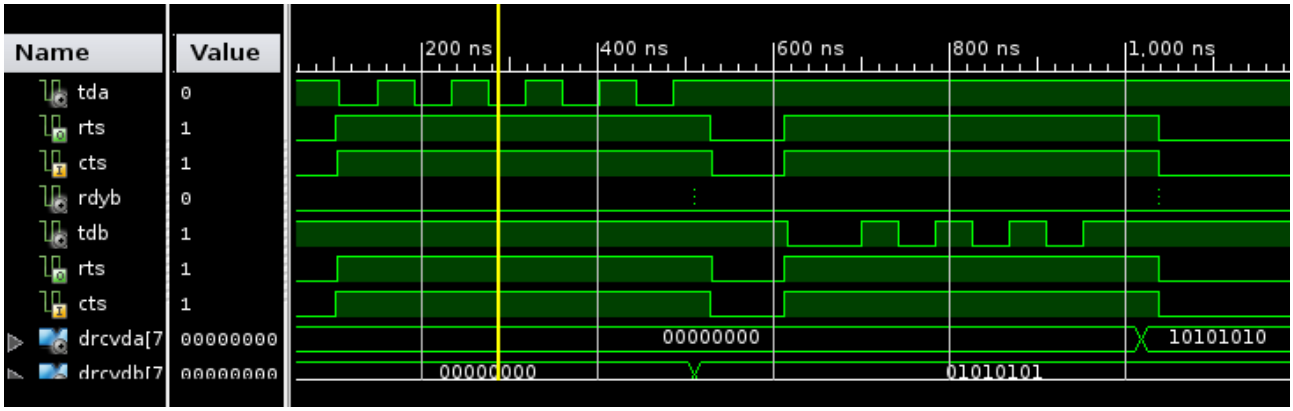
12 The modem acknowledges this.

13 DTR is dropped by the computer; this causes most modems to hang up. After hang-up, the modem acknowledges with DSR low. If the connection breaks, the modem also drops DSR to inform the computer about it.

En el siguiente diagrama en bloques se observa globalmente la interconexión de los sistemas implementados.



Además, se adjunta una captura de la simulación realizada.



En donde se observa que primero envía el sistema A y luego lo hace el sistema B

Referencias:

- en.wikipedia.org/wiki/Transputer
- ict.udlap.mx/people/oleg/docencia/PARALELO/lib22.html
- es.wikipedia.org/wiki/Occam_%28lenguaje_de_programaci%C3%B3n%29
- www.geocities.ws/mofern/PaperOccamJAIIO_s.htm
- www.google.com.ar/url?sa=t&rct=j&q=&esrc=s&source=web&cd=10&ved=0ahUKEwiO6sGPmqXUAhUKE5AKHUR3BPgQFghXMAk&url=http%3A%2F%2Fxavier.fenard.free.fr%2Ffichiers%2F010_20130612_Transputer-Architecture_Handout_UM.pdf&usg=AFQjCNF11RpQU_K1kt-82k0_4Hdt-iXKUQ&sig2=cIahOvUJc-9dE-Q2BrTwMw&cad=rja
- www.transputer.net/tn/tn.asp Notas de aplicación...
- http://www.google.com.ar/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&ved=0ahUKEwil6v_kraXUAhVGQpAKHWaWCPYQFggwMAI&url=http%3A%2F%2Fwww4.di.uminho.pt%2F~jmf%2FPUBLI%2Fpapers%2F1997-ieecdt.pdf&usg=AFQjCNF7toCXSXGt50mXGdOL366xyNt8-w&sig2=LBXtXTkpOcOVklmBEfN4yw&cad=rja
- http://www.google.com.ar/url?sa=t&rct=j&q=&esrc=s&source=web&cd=4&ved=0ahUKEwil6v_kraXUAhVGQpAKHWaWCPYQFgg6MAM&url=http%3A%2F%2Fgec.di.uminho.pt%2Fmethodes%2Farticle3.pdf&usg=AFQjCNHPiD85LcxOov06zWvcpAp7eZDIUA&sig2=90IFgA1cIgm4NKKv0bhnew&cad=rja
- Communicating Process Architectures 2004 - WoTUG-27
- http://www.google.com.ar/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&ved=0ahUKEwjM0uigwqrUAhUP32MKHVyPBLQQFggsMAE&url=http%3A%2F%2Fwww.comp.tmu.ac.jp%2Fmorbier%2Fwork%2FTPCORE.pdf&usg=AFQjCNGbkRV4JgI0uwmkrDnxeG0YYeFUVA&sig2=IJG7COhsZWaaNwx2xmPmmg
- Iván Esteben Lopetegui Téllez – Tesis de grado “Diseño de modelos de sistemas para diagnosticar fallas usando métodos formales”
- http://www.google.com.ar/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0ahUKEwjgh96q07DUAhUEj5AKHcsQASYQFggoMAA&url=http%3A%2F%2Farantxa.ii.uam.es%2F~ivan%2Fst4-jcra2003.pdf&usg=AFQjCNGZMhT_y6bYMjwrlQqYLf-iSRdj-g&sig2=hQSSMQLcdnFgIyP6rZ-7CA&cad=rja