

Laboratorio 01 - CLAVE

Competencias para desarrollar

Aplicar los conceptos relacionados a fundamentos de microprocesadores y su funcionamiento.

Nelson García Bravatti - 22434

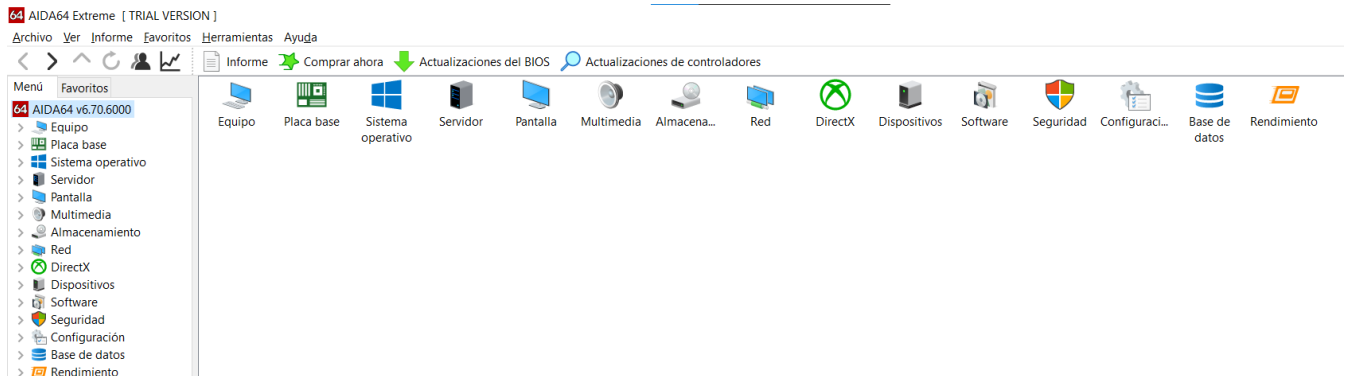
Instrucciones

Esta actividad se realizará individualmente. Solamente se permitirá trabajar la segunda parte de este laboratorio en parejas, en casos donde sea comprobable que el software utilizado no es compatible con el sistema operativo del ordenador del estudiante. Al finalizar los períodos de laboratorio o clase, deberá dejar constancia de sus avances en Canvas, según indique su catedrático.

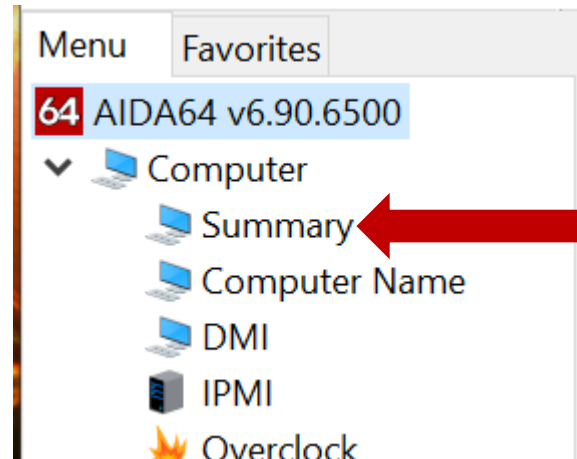
Parte 01: conociendo su equipo:

1. Verifique las características técnicas del procesador en su equipo utilizando el software AIDA64 y complete las tablas con la información solicitada.

- Abra el siguiente enlace <https://www.aida64.com/downloads> en el navegador de su preferencia.
- Descargue el software AIDA64 que mejor se adapte a las especificaciones técnicas de su equipo.
- Descomprima el .zip descargado, y ejecute aida64.exe. Esta es una versión gratuita de prueba de 2-4 semanas, por lo que no tendrá todas las funcionalidades habilitadas. El ejecutable mostrará la siguiente ventana (los iconos y opciones variarán según las características de su equipo):




- Seleccione la pestaña *Equipo* y la opción *Resumen*, localizada en el menú principal.




e. (10 puntos) Complete los siguientes datos:

Tipo de equipo	Equipo basado en x64 ACPI (Mobile)
Sistema operativo	Microsoft Windows 10 Home Single Language
Tipo de CPU	QuadCore Intel Core i7-8565U, 2000 MHz
Nombre de motherboard	Dell Inspiron 5480
Chipset de motherboard	Intel Cannon Point-LP, Intel Whiskey Lake-U
DIMM1 (si hubiese)	SODIMM DDR4 8 GB
DIMM2 (si hubiese)	DIMM B
DIMM3 (si hubiese)	-
Aceleradoras 3D	Intel UHD Graphics 620
Aceleradoras 3D	nVIDIA GeForce MX150

f. (10 puntos) Identifique qué significan las palabras y valores alfanuméricos en el nombre del microprocesador de su PC, que se muestran en CPU Type.

 Motherboard

 CPU Type

QuadCore Intel Core i7-1165G7, 4000 MHz (4

Marca	Intel
Producto	QuadCore Intel Core i7-8565U
Gama	Gama alta, U
Identificación de generación	8ª generación de procesadores Intel® Core™ i7
SKU (pueden ser varios)	2902 0000
Sufijo de producto	QuadCore i7

g. Presione *doble click* sobre la opción CPU Type, para conocer más sobre las especificaciones técnicas de su procesador.

En este espacio encontrará *propiedades de la CPU, fabricante de la CPU* (en esta sección encontrará el enlace a la hoja de datos del microprocesador, puede usar esta para completar la información solicitada), *Multi CPU*, y *Utilización de la CPU*.

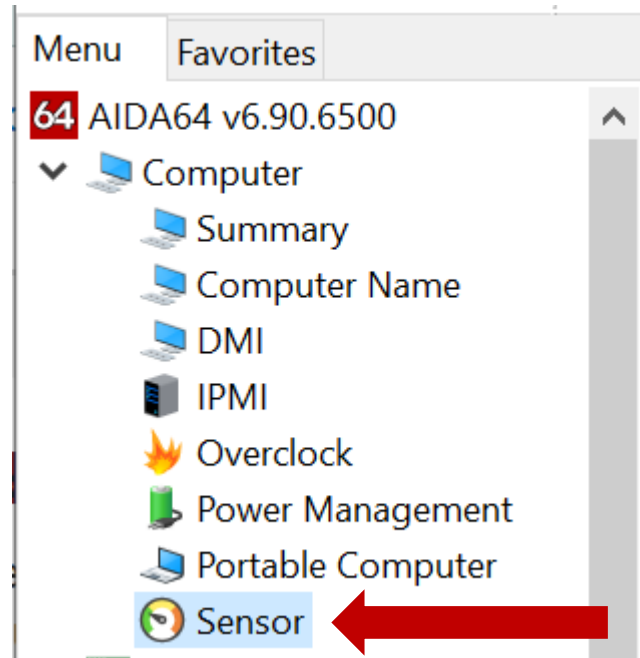
h. (20 puntos) Complete la siguiente información

Tipo de CPU	QuadCore Intel Core i7-8565U, 2000 MHz
Alias de la CPU	Whiskey Lake-U
Conjunto de instrucciones	x86, x86-64, MMX, SSE, SSE2, SSE3, SSSE3, SSE4.1, SSE4.2, AVX, AVX2, FMA, AES
Caché de código L1	32 KB per core
Caché de datos L1	No se puede observar
Caché L2	256 KB per core (On-Die, ECC, Full-Speed)
Caché L3	8 MB (On-Die, ECC, Full-Speed)
Cantidad de Núcleos	4
Cantidad de Cores	4
Cantidad de Threads	8
Litografía	14 nm
Max Turbo Frequency	4.60 GHz
Processor Base Frequency	1.80 GHz
Cache	8 MB Intel® Smart Cache
Bus Speed	4 GT/s
TDP	15 W
T_{JUNCTION}	100°C
Sockets Supported	FCBGA1528
Package Size	46 mm x 24 mm
Tecnologías soportadas	3

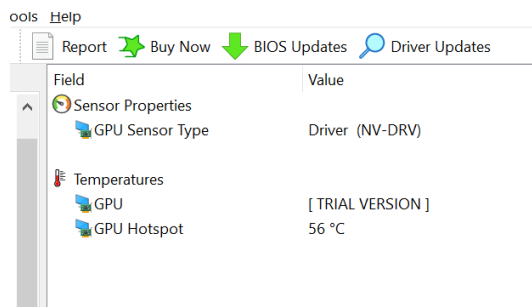
i. (05 puntos) Observe el comportamiento de la utilización de la CPU abriendo programas que generan una mayor exigencia en el desempeño de su equipo. Explique cómo se comporta el porcentaje de utilización de cada core/thread.

R// Al aumentar el número de programas en ejecución el CPU necesita un mayor desempeño, por lo cual utiliza más cada core y thread, aumenta el porcentaje de utilización de cada uno, si se le resta al número de programas en ejecución el CPU ya no necesitará utilizar con más intensidad los cores o threads.

2. Seleccione la pestaña *Equipo* y luego la opción *Sensor*, ubicada dentro del menú principal.



- a. (05 puntos) Interprete y explique qué información encuentra en *Sensor*.



R// Se indica que el tipo de sensor del GPU es Driver (NV-DRV) además, indica la temperatura actual de la computadora

- b. (02 puntos) ¿Cuál es el rango de temperatura que poseen los núcleos?

R// Pueden soportar hasta una temperatura de 100°C.

- c. (03 puntos) ¿Los valores visualizados se encuentran dentro de los valores permitidos de temperatura? Explique.

R// Si, ya que, no supera las 100°C de temperatura.

Parte 02: ejecución de procesos.

Puede utilizar <https://www.onlinegdb.com/>. Seleccionar lenguaje c++.

3. Ejecute el contenido del archivo Programa_1.cpp. Observe que en el programa principal se encuentra una estructura if-else.

- a. (05 puntos) ¿A cuántas de las opciones de la estructura if-else se ingresa al ejecutar el programa?

R// 3

b. (05 puntos) Explique por qué se obtiene el resultado obtenido.

```
main.cpp
15 {
16     pid_t pid;
17
18     printf("    --- PARENT PROCESS STARTED ----\n\n");
19
20     pid = fork();                /*Fork a child process*/
21
22     if (pid < 0) {                /*Error occurred */
23         fprintf(stderr, "WARNING!! Fork failed");
24         return 1;
25     }
26     else if (pid==0) {            /*Child process */
27         printf("This is the CHILD process - STATUS: CREATED\n");
28     }
29     else {                        /*PARENT PROCESS */
30         /*Parent will wait for the child to complete */
31         printf("This is the PARENT process - STATUS: RUNNING\n");
32     }
33 }
34 }
```

input

```
--- PARENT PROCESS STARTED ---
This is the PARENT process - STATUS: RUNNING
...Program finished with exit code 0
```

R// El código emula la creación y ejecución de procesos, el resultado indica que a la hora de crear un nuevo proceso este no falla, entonces comienza la ejecución del proceso padre, no sin antes verificar que haya algún proceso hijo que se esté ejecutando, cuando el hijo termine el padre puede comenzar a ejecutarse.

4. (10 puntos) Ejecute Programa_2.cpp. Indique qué representan los valores ID, impresos desde el *Parent Process* y desde el *Child Process*.

```
main.cpp
18 if (var == 0){
19     // valor de sleep para child es 10                //Child process, return 0
20     sleep(10);
21     printf("Hello from Child!\n");
22     printf("Fork value from Child is: %d\n", var);
23 }
24 else {
25     // valor de sleep para parent es 10                // pid != 0.
26     sleep(10);
27     printf("Hello from Parent!\n");
28     printf("Fork value from Parent is: %d\n", var);
29 }
30 }
31
32 int main()
33 {
34     rutina();
35     return 0;
36 }
37 }
```

input

```
Hello from Parent!
Hello from Child!
Fork value from Parent is: 40
Fork value from Child is: 0
```

R// En este caso el valor 0 para el 'Child' que el proceso fue creado y ejecutado, para cualquier valor de 'Parent' este indica que fue creado, pero si se da el caso esperará a 'Child' para empezar a ejecutarse. Si no aparece 'Child' esto indica que no ha sido creado.

5. Ejecute Programa_3.cpp.

a. (03 puntos) ¿Qué valor posee x al imprimirlo desde el Parent Process?



```

main.cpp
12 #include <sys/wait.h>
13
14 void rutina()
15 {
16     pid_t pid;
17     int status;
18     pid = fork();
19     int x = 10;
20
21     if (pid == 0) {
22         printf("Child: Variable value x = %d\n", ++x);
23         _exit(0);
24     }
25     else {
26         printf("Parent: Variable value x = %d\n", --x);
27         (void)waitpid(pid,&status,0);
28     }
29 }
30
31 int main()
32 {
33     f

```

input

```

Parent: Variable value x = 9
Child: Variable value x = 11
...Program finished with exit code 0
Press ENTER to exit console.

```

R// 9

b. (03 puntos) ¿Qué valor posee x al imprimirlo desde el Child Process?

R// 11

c. (09 puntos) Explique ampliamente ¿por qué se obtienen estos valores de x? ¿x es una variable compartida?

R// X no es una variable compartida, cuando se asigna pid con la función fork() se crean dos procesos idénticos entonces a la hora de que se modifique x en cada uno de ellos la variable es modificada, pero esta modificación no afecta la variable del otro proceso, es por eso que al final cada uno imprime un valor diferente.