Instrucciones generales para la realización de este examen

La respuesta debe escribirse en el hueco existente a continuación de cada pregunta con **letra clara**. Cada respuesta incorrecta, ilegible o vacía no suma ni resta. En el caso de preguntas teóricas se valorará la capacidad de síntesis.

---

1 ❑ Se ejecuta un benchmark multihilo, con 3 hilos, en dos computadores diferentes, A y B. El computador A tiene un procesador de 4 núcleos mientras que el procesador B es mononúcleo. El tiempo medio de respuesta del benchmark es de 15.7 segundos en el computador A y de 31.5 segundos en el computador B.

a — (0.25 puntos) ¿Cuál es el rendimiento de cada computador utilizando la productividad?

A:	B:

b — (0.5 puntos) ¿Cuál es la aceleración del computador A con respecto al computador B?

c — (0.75 puntos) Se sutituye el procesador del computador B por uno multinúcleo con 4 núcleos. El tiempo de respuesta del benchmark se divide por el número de hilos que se pueden ejecutar de forma concurrente en el sistema. En este nuevo escenario, ¿cuál es la aceleración del computador B con respecto a A?

2 ❑ Sea una CPU segmentada con pipeline de 10 etapas y una productividad maxima de 400 MIPS.

a — (0.5 puntos) ¿Cuál es la frecuencia de esta CPU, expresada en GHz?

b — (0.5 puntos) ¿Cuál debería ser el ancho de emisión de esta CPU si se decidiera convertirla en superescalar para lograr una productividad máxima de 2800 MIPS?

3 ❑ Se ejecuta el siguiente programa MIPS64 sobre una microarquitectura segmentada básica cuya única mejora es una unidad no segmentada de multiplicación/división de enteros de 5 ciclos. Las excepciones precisas no están soportadas.

```
1      xori r4, r0, 1 ; r4 = 1
2      slt r2, r3, r4
3      bne r4, r0, label ; branch on r4 not equal to r0
4      dsub r3, r9, r7
5  label:
6      dmul r5, r9, r6
7      ori r5, r0, 7
```

a — (1 punto) Enumera las dependencias de datos que existen en el programa, especificando tipo junto con las instrucciones y registros involucrados. **Ejemplo de respuesta**: RAW => `daddi, ori : r7` // `dsub, xori : r3`

RAW =>

WAW =>

WAR =>

b — (1 punto) Indica las duraciones de las detenciones expresadas en ciclos de reloj. Para los tipos RAW, WAW y estructural debe indicarse además la instrucción que se detiene, mientras que para el tipo control la instrucción que la provoca. Si un tipo de detención no aparece, indícalo con **N/A**. Si hay varias detenciones del mismo tipo deben indicarse todas.
Ejemplo de respuesta, RAW: `xor` 1 ciclo // `andi` 2 ciclos

RAW:

WAW:

Estructural:

Control:

c — (1 punto) ¿Cuántos ciclos de reloj son necesarios para ejecutar el código anterior? ¿Cuál es el CPI ignorando el transitorio inicial? **Indica la expresión matemática** utilizada para obtener el CPI.

Ciclos:	CPI:

d — (0.5 puntos) Si todas las rutas de reenvío estuviesen implementadas en esta microarquitectura, ¿qué rutas se activarían al ejecutar el programa anterior? **Ejemplo de respuesta**: Salida `EX daddi` –> Entrada `EX dmul`.

e — (1 punto) Si se implementase renombrado de registros en esta microarquitectura, ¿qué registros arquitectónicos cambiarían de registro físico al final de la ejecución del programa y a qué registro físico estarían asignados? Para el renombrado se dispone de una cola FIFO que originalmente contiene los registros `rr32` a `rr63` y a los que se van añadiendo los registros disponibles
Ejemplo de respuesta para esta pregunta: Registro `r7` asignado a `rr43`.

4 ❑ Una microarquitectura MIPS64 implementa **evaluación clásica de saltos** (en la etapa `MEM`), predición de saltos siempre no tomado y **sin rutas de reenvío**. Sobre esta microarquitectura se ejecuta el siguiente código.

```
1         daddi   r9, r0, 648
2         daddi   r7, r0, 33
3  startloop:
4         daddi  r8, r15, r10
5         daddi r9, r9, -1
6         beqz  r9, exitloop  ; Branch on Equal Zero
7         daddi r7, r7, -3     ; r7 decrement by -3
8         bneqz r7, startloop  ; Branch on NOT Equal Zero
9  exitloop:
10        sd    r8, 100(r0)
11        xor   r8, r8, r8
```

a — (1.5 puntos) Rellena **todas las filas** del siguiente cronograma con la evolución de las primeras instrucciones del programa sobre el *pipeline*.

Instrucción \ Ciclo	1	2	3	4	5	6	7	8	9	10	11	12	13	14
`daddi r9, r0, 648`	IF													
`daddi r7, r0, 33`														
-														
-														

b — (0.5 puntos) Teniendo en cuenta las dos instrucciones de salto del programa. Para el programa completo, ¿Cuántas veces en total acierta la predicción el predictor siempre no tomado? ¿Y cuántas veces en total falla la predicción? Nota: el predictor solo se aplica a las instrucciones de salto condicional.

Núm. aciertos:	Núm. fallos:

c — (0,5 puntos) Si el predictor de saltos siempre no tomado se sustituye por un predictor dinámico de 2 bits con el valor `01` por defecto del historial (*weak not taken*), ¿cuántos ciclos de reloj se detendría el *pipeline* en la ejecución del **programa completo** debido a la instrucción `beqz r9, exitloop` y a la instrucción `bneqz r7, startloop`?

`beqz r9, exitloop` :	`bneqz r7, startloop`:

d — (0,5 puntos) Si el programa se ha cargado a partir de la dirección de memoria `C7ECh` ¿Cuáles serán los valores de las entradas en la tabla BHT+BTB correspondientes a las instrucciones de salto al finalizar la ejecución del programa?

Dirección	Destino	Historial