SDC Assignment Report

Task 1

Q1. The obtained latency is always minimal since there are no resource constraints. The latency is determined solely by data dependencies. At any given moment all ready-operations are scheduled. The solution is unique.

Q2. The scheduling problem should be solved for each BB individually because the BBs are executed sequentially as stated in the task description. If we scheduled the CDFG as a whole, some BBs would have overlapping schedules, thus violating this requirement. By doing this, the complexity of the problem is reduced significantly.

Task 2

Q1. Since I used the tip from remark a) the ALAP latency is identical to the ASAP latency. Slack describes the mobility of an operation. It is defined as the difference of starting times between ASAP and ALAP. It can be used as a priority measure for resource constrained scheduling.

Q2. It depends on the kernel. For BB0 in kernel_1, ASAP results in 4 concurrent memory accesses in cycle 0. Thus, ALAP would be favorable since it requires less area for the same performance. For kernel_2, ALAP is favorable for the same reason as kernel_1.

Task 3

Q1. Assuming sequential execution of each BB:

Area (mul, add, zext)	Latency Kernel 1	Latency Kernel 2
3 (1,1,1)	25	26
*	25	26

Area (mul, add, zext)	Latency Kernel 3 (one iter.)
3 (1,1,1)	18
6 (2,2,2)	15
9 (3,3,3)	14
4 (2,1,1)	18
4 (1,2,1)	17
4 (1,1,2)	16
5 (1,1,3)	16
5 (1,2,2)	15
6 (1,3,2)	14
6 (1,2,3)	15

Area (mul, add, zext)	Latency Kernel 4 (one iter.)
3 (1,1,1)	25
6 (2,2,2)	24
9 (3,3,3)	24
4 (1,2,1)	24

= pareto-optimal configuration

Q2. See pareto-optimal configurations in green; they provide the best latency considering the required area and you cannot achieve the respective latency with less area. For kernel 1, 2 are heavily bottlenecked by their data dependent critical path. Hence, increasing resource constraints won't enable faster scheduling. For kernel 3, sweeping configurations provide more interesting results as additional adders and zero extensions can decrease the latency. Lastly, kernel 4 profited from having 2 adders.

Task 4

Q1. $L_{tot} = L_{entry} + L_{loop.body}+II*(N-1) + L_{loop.end}$

Q2. Kernel_3: II = 2, Kernel_4: II = 19

Task 5

Q1. No, because operations are scheduled from their topological order which doesn't necessarily yield the best latency. It's only a heuristic approach and thus can't guarantee the best solution.

Q2. The budget: setting it too low might result in a failed schedule early on although there might have been a feasible solution with the current II. However, a huge budget can lead to a poor performance.

Q3. The order in which operations are scheduled can be decided by a different measure such as slack or the distance to the sink.