A1:

100%, weil erst p0 und anschließend p1 abgearbeitet wird und die cpu dardurch nie warten muss

A2:

11 Zeiteinheiten ,

erst 4 Zeiteinheiten run p0,

1 Zeiteinheit run IO p1,

5 Zeiteinheiten warten,

1 Zeiteinheit io\_done

A3:

7 Zeiteinheiten, weil p1 bearbeitet wird, während p0 wartet

A4:

Es dauert wieder länger, da p1 nicht gestartet wird während p0 wartet

A5:

Braucht nur noch 7 Zeiteinheiten, da p1 wieder bearbeitet wird während p0 wartet – scheint standartverhalten zu sein

A6:

Io 1 wird erst beendet, nachdem p1 – p3 fertig sind, dardurch bleibt io lange ungenutzt. Wenn io2/3 ausgeführt werden bleibt hingegen die cpu lange ungenutzt.

A7:

CPU usage 100%, da p1 bis p3 ausgeführt werden während p0 wartet

Gute idee, weil cpu und io parallel genutzt werden

A8:

s3

IO\_RUN\_IMMEDIATE 17 : IO\_RUN\_LATER 18

Letzte cpu instruction von p0 wird aufgeschoben und bearbeitet während p1 wartet

SWITCH\_ON\_IO 18 : SWITCH\_ON\_END 24

Großer teil von p1 wird erledigt während p0 wartet

Mit IO\_RUN\_IMMEDIATE von 18 auf 17 Zeiteinheiten, da letzte cpu instruktion erledigt wird während p1 wartet

IO\_RUN\_LATER und SWITCH\_ON\_END verlangsamen die berarbeitung immer