(Gruppe Özcan, Bahr, Portscher)

Aufgabe 2:

Was macht der Code?

Im Code laufen 2 parallele Threads. Der Master-Threads soll folgende Schreiboperationen auf Variablen ausführen:

data = 42
flag = 1

Ein Thread mit der Threadnummer 1 soll warten, bis flag auf 1 gesetzt wurde und daraufhin die beiden Variablen flag und data ausgeben.

Wird das Programm 1000 Mal ausgeführt, gibt es immer “flag=1 data=42” zurück. Allerdings wird das nur aus dem Code heraus nicht gerantiert. Dazu ist die Flush-Operation da.

Flush sorgt für eine konsistente Sicht auf den Speicher.
#pragma opm flush [(list)] markiert den Punkt, an dem einem Thread garantiert wird, dass seine Sicht auf dem Speicher (im Bezug auf die Variablen in der Liste) konsistent ist. Alle bisher getätigten Lese oder Schrieboperationen des Threads wurden beendet und wurden im Hauptspeicher gesichert. Diese konsistenzen beziehen sich auf Operationen zwischen dem Thread und dem Hauptspeicher.

Benutzung von Flush in diesem Fall: siehe c-File

So wird auf jeden Fall garantiert, dass die Schreiboperation auf die Variable flag von Thread 0 im Thread 1 sichtbar und richtig ist.

Der Code in flush.c beschreibt eine typische Producer/Consumer-Situation. Der Compiler kann potenziell die Lese- und Schreiboperationen auf data und flag umordnen (Die -O3 Flag beim Kompilieren optimiert den Code stark, daher ist das sogar recht wahrscheinlich.)