CSCI 360 Assignment 4: ABENDs and Dump Reading
David Flowers II z1942130

1. What is the address of the next instruction which would have been executed?
000030

2. What is the value of the condition code at the time of the ABEND?
Condition code was 10 or 2 in decimal, meaning the result was positive.

3. What is the length of the instruction that caused the ABEND (number of bytes)?
It is 10 which is 2, then * 2 so it's a 4-byte long instruction.

4. What does the value stored at location counter value 00002C represent?
So the numerical value is 1344335922, but it gets encoded as 5020F032 which encodes an instruction that probably shouldn't be there. The instruction is ST 2,50(,15) which is probably not what the programmer intended, and might explain why it ABENDS at storage Op code after abend

5. What is the address of the instruction that caused the abend?
000030 – 000004 = 00002C

6. What type of error occurred (number and name)?
It was a S0C6 - Specification Error

7. What usually causes this error?
Loading storage that's not on a fullword boundary.

8. What does the value in register 7 represent at the time of the ABEND dump?
Its supposed to be the value of the addition going on and then stored into SUMMED, but it also looks like an encoded instruction. It seems to be a BR 15,44(,15)

9. What instruction needs to be added to fix this ABEND?
We forgot the BCR B'1111',14 before the LTORG instruction.

10. Did this error occur while the program was being assembled or when it was being run?
While it was being run

11. What exactly happened here to cause this ABEND? Be detailed but succinct in your description
So it seems that in the program storage past the LTORG that an instruction got encoded into the value. This instruction stored R2 into a value that had a wrong displacement. The displacement in the instruction was 50, 50 is not divisible by 4 (a fullword length) and this you get the specification error, i.e. loading storage that's not a fullword boundary. The ABEND is pointing towards this S0C6 but the issue actually was that we were missing a BCR instruction, and after Register 7s addition, encoded a BR instruction which branched, I'm assuming, to the accidental

encoded storage that was not on a fullword. And because the BCR was not there, the program continued to run instructions that were encoded in storage.