We first use a cross-compiler to compile a C program into .o file this is done by a GCC compiler, now we convert this object into to .coff by linking this to a start.o. Now this object is converted into a noff by a Nachos default function coff2noff\*, now this object is a nachos object and is nachos executable. halt.o: halt.c $(CC) $(CFLAGS) -c halt.c halt: halt.o start.o $(LD) $(LDFLAGS) start.o halt.o -o halt.coff ../bin/coff2noff halt.coff h First line of code given on handout is conveying that we are creating .o file which is depending on the .c this is standard way of stating this with a colon so this means the same wherever this format is used. Then we are creating variables identified by $ mark CC = $(GCCDIR)gcc -B../../../gnu-decstation-ultrix/ CFLAGS = -G 0 -c $(INCDIR) LD = $(GCCDIR)ld LDFLAGS = -T script -N GCCDIR = ../../../gnu-decstation-ultrix/decstation-ultrix/2.95.3/ CC define that we are using GCC compiler to create .o file and –c defines not to run the linker. CFLAGS represents that compilation will be with same value of –G, LD and LDFLAGS specifies the linker I,e GNU linker along with a path is provided for it. LDFLAGS has –T which defines not to use default script for referencing by linker where –N is to set data and text read/writeable.