היי משה.

שלום אולגה ויואב,

מה נשמע משה

אנחנו יושבים על הפרוייקט ועולות לנו כמה שאלות קונקרטיות:

1. אנחנו מתמקדים ב Display controller ומבינים מה הקווים שמגיעים מבחוץ. אבל יש לנו בעיה עם שני קווים:

א. wbs\_tga\_i [9..0]. k  מה בדיוק הוא נותן לנו ? איזה burst זה ? מה זה מייצג?

באופן כללי, בכל ממשקי הווישבון, בארי הוסיף קו tga [9..0] אשר מייצג את אורך הטראנזקציה, כלומר, כמה מילים (bytes) יועברו באותה טראנזקציה (כמות ה- stb ב- cyc).

ספציפי, ב- display controller, ה- wbs\_tga\_i, מייצג את אורך הטראנזקציה (למען הדיוק – ה- TGA תמיד שווה לכמות המילים שיועברו פחות 1. כלומר, כדי לשדר 10 מילים, אז הערך של ה- TGA הזה יהיה 9), מה- host (דרך ה- rx path וה- interconn) ל- display controller, במטרה לכתוב לרגיסטרים שבתוכו.

ב. wbs\_tgc\_i  מה עושים עם זה? זה נראה לנו כאילו זה סיגנל שכל משתמש יכול להשתמש בו בצורה שונה לצרכיו שלו . אנחנו טועים?

ה- TGC הוא ביט אחד, ובמימוש של בארי, הוא נועד לקבוע האם המידע נרשם לרגיסטרים של אותו בלוק או לא (ומה זה לא – לא יכול להיות לעוד קליינט שיושב על אותו ווישבון. כלומר, הקו הזה יכול להפריד האם על התעבורה של הווישבון תועבר לקייינט X או לקיינט Y). הדבר לא רלוונטי בבלוק ה- display controller, כיוון שרושמים אליו רק לרגיסטרים ולא לעוד בלוק בתוכו. כדי לדחוף את ה- VESA הוא משתמש בממשק ה- WBM ולא ה- WBS ולכן אין שני בלוקים פנימיים שיושבים על אותו ממשק ווישבון באותו בלוק. דוגמא להיכן הסיגנל הזה כן רלוונטי היא ב- memory management. שם ה- rx path יכול לרשום הן ל- mem wr cntrl והן לרגיסטרים של הבלוק, כלומר, שני קליינטים יושבים על אותו ממשק ווישבון, ובאמצעות סיגנל בקרה זה קובעים למי המידע שמגיע דרך ה- host יועבר.

צירפנו לך את המסמך של ה display controller ואנחנו מדברים שם על הסיגנלים היוצאים ונכנסים מתוך

**Display Controller TOP Pinout**

דבר נוסף וקריטי:

לפי חישוב שלנו טעינו בחישוב של אורך ה opcode : זה לא שאורך הaddress הוא 19 אלא רק 12 כמספר ה 12^2 שורות שב SDRAM אבל לא צריך bit אחרון ולכן מספיק לנו רק 11

 בנוסף גם נראה לנו שצריך להגיד מאיזה bank מתוך ה-4 רוצים לקרוא ולכן מפה לשם אנחנו צריכים opcode של 23 bit.

השאלה היא כזו: במסגרת חבילת שינויים נרצה לדעת מהו השינוי האחרון: האם כדאי להשתמש בביט נוסף בopcode שייצג לנו האם זהן ה opcode (השינוי ) האחרון ואז יושב לנו בול  opcode באורך 24 סיביות או לחילופין להשתמש בעובדה שבתוך  הפקודה הגדולה שמקבלים מהמטלב     (SOF,TYPE,ADR,DATA LENGTH, ... EOF )  נשתמש איכשהו  ב DATA LENGTH   (עוד לנו ברור לנו עדיין איך נחלץ אותו בדיוק)

במימוש של בארי, ה- memory controller מקבל 22 ביטים של כתובת ל- sdram. אין צורך לחסוך. התוכנה הרי מחשבת את הכתובת ולא החומרה. ננסה להבהיר כי יש פה אי הבנה - ראשית ה- RAM (שקף 13 במצגת האפיון) שמחזיק את טבלת המיפויים יאותחל לאפס בזמן העלייה של המערכת. ה- USER מקליד מה הוא רוצה לראות בפעם הראשונה. התוכנה (ב- matlab) אמורה לעשות את מירב העבודה. נניח שה- USER ביקש 3 אייקונים במקומות שונים בפעם הראשונה. התוכנה מכינה את חבילת המידע בהתאם לאייקונים והמיקומים. על כל שינוי יש לשלוח 4 בתים (מיקום ב- SDRAM של האייקון, מיקום בתצוגה, והאם הורדה או הוספה – שקף 11 במצגת האפיון). כלומר, חבילת מידע בעלת 12 בתים + כל ה- HEADER וה- TAIL. כל שלושת ה- 4 בתים (שלושת ה- OPCODE-ים) מגיעים בשדה ה- DATA ולא בשום שדה אחר, לכן לא ברור לי מה רשמתם. ה- MSB של כל 4 בתים (כל OPCODE) יהיה '1' (סימן להוספה ולא הורדה של אייקון). בחומרה אצלכם, מכל 4 בתים ישלף ויחושב המיקום ב- RAM שאליו מיועד להרשם כתובת ה- SDRAM שבה ממוקם האייקון, וסיגנל הבקרה האם הורדה/הוספה. נמשיך עם הדוגמא. נניח שבפעם השנייה, ה- USER מבקש להוריד אייקון אחד שהוא בחר בפעם הראשונה ולהוסיף אייקון אחר במיקום אחר. הפעם תשלחנה חבילת מידע בעלת 8 בתים (כל OPCODE 4 בתים), כאשר עבור ההורדה, ה- MSB של ה- 4 בתים יהיה '0' (סימן להורדה ולא הוספה).

אנחנו חושבים שבאמת משהו לא ברור כאן:

קודם כל רצינו להבהיר משהו בנוגע לקריאה וכתיבה מה SDRAM:

כשאנחנו פונים ל memory management top אז יש לנו בסך הכל 10 סיביות להפנות ל address של ה sdram דרך הקו Wr\_wbs\_adr\_i[9..0] לצורך כתיבה (כלומר אנחנו פונים ל mem\_cntr\_wr ומשם כותבים ל sdram) ובאותו אופן אנחנו פונים לקריאה מה sdram בעזרת 10 סיביות בקו Rd\_wbs\_adr\_i[9..0] שמפנות לaddress של ה sdram (הפעם דרך ה ( mem\_cntr\_rd.

לא כל כך הבנו איך אנחנו יכולים לתת במכה 22 סיביות לצורך כתיבה / קריאה מה sdram והרי זה מה שצריך לתת לו (האם ה22 מתחלקים בתזמונים שונים איכשהו)?

בנוסף, אנחנו רוצים להבהיר משהו:

אנחנו לא צריכים שב opcode יהיו 4 בתים שייצגו את השינוי של סמל מסויים אלא רק 23 ביטים שמיוצגים ע"י 3 בתים וזאת כיוון שכל סמל יושב בתחילת כל שורה זוגית ולכן אין צורך לא במיקום של col ולא בסיבית ה LSB של ה row כיוון שהיא תמיד 0.

אנחנו כן נשלח לsdram את כל ה 22 ביט אבל את זה כבר נעשה מהבלוק שלנו שיידע לקדם את ה col .

דבר נוסף:

אם נלך לפי הדוגמא שלך ונניח שבמסגרת חבילת השינויים יש לנו 3 שינויים, איך אנחנו נדע שיש 3?

אנחנו חשבנו על 2 אופציות:

1. ה user שולח את אורך ה data בפקודה "הגדולה" (SOF,TYPE,ADR,DATA LENGTH, ... EOF ) ואנחנו יכולים להשתמש בשדה DATA LENGTH (עוד לא ברור לנו איך נחלץ אותו).
2. לחילופין, אמרנו שיש 23 ביט בopcode ולכן חשבנו שבopcode השלישי נציב 1 בסיבית ה24 ובשניים האחרים נציב 0 וכך נדע שהגענו לשלישי ולאחרון.

מה אתה אומר?

אגב, אם אתה אומר שלא צריך את כל הbanks אז חסכנו אפילו עוד 2 סיביות (נניח אחנו משתמשים רק בbank אחד ידוע מראש) ואז מ23 סיביות של opcode ירדנו ל 21.

בתכלס אנחנו לא בטוחים עדיין אם יש לנו צורך בכל ה banks או שנשתמש רק באחד, ואז נחסוך עוד כמה סיביות מה opcode.

אין צורך בכל ה- banks, אך אני מקווה שהבנתם את הפסקה הקודמת שרשמתי. אם לא – נא לשאול.

בברכת חג שמח וסופ"ש נעים - יואב ואולגה

חג שמח,

משה

תודה וערב טוב!