# Sinus und Cosinus Funktionen

Die Ergebnisse der einzelnen Sinus bzw. Cosinus Berechnungen, für die Gradzahlen von 0-90 sind im Memory hinterlegt. Über Label „.sin\_lookup“ kann man auf die Startadresse zugreifen. Alle 4 Bit liegt einer neuer Sinus.- Cosinus Wert im Speicher. Möchte man nun auf einen Speziellen Sinus.- Cosinus Wert zugreifen, muss man die Gradzahl mit 4 multiplizieren und das Ergebnis auf die Startadresse addieren. Das Ergebnis ist die Adresse an dem der gesuchte Wert hinterlegt ist.

# Die draw\_pixel Funktion

Die draw\_pixel Funktion, speichert den Hexadezimal Wert von einer Farbe an eine gewünschte Adresse im Speicher. Die Adresse ermittelt sich aus dem X.- und Y.- Koordinaten, die der Funktion über die a.-Register übergeben werden. Das Bitmap-Tool wertet die Adressen aus und stellt den Inhalt grafisch dar. Das Bitmap-Tool wird auf 256 \* 256 Pixel eingestellt. Die Richtige Adresse wird aus den X.- und Y.- Koordinaten wie folgt berechnet.

Für die Berechnung werden nur Temporäre Register benutzt. Somit müssen keine Register mittels Callee Save gesichert werden, da Temporäre Register durch Caller Save gesichert werden.

# Kreis mit dem Float-Algorithmus zeichnen

Der gegebene C-Code bot eine gute Orientierung. Wir haben für die Aufgabe hauptsächlich S-Register verwendet, um ein Caller-Safe zu vermeiden. Unsere Unterprogramme sind nicht sehr umfangreich und benötigen keine S-Register, und somit kein Callee Safe. Demzufolge mussten wir in unserem Programm so gut wie keine Stack Operationen verwenden. Wir haben erst alle Tests bestanden, nachdem wir die Integer to float Unterprogramm ausgeführt haben.

# Aufgabe 2. Kreise mit dem Integer Algorithmus zeichnen

Die zweite Aufgabe verlief deutlich schneller als Aufgabe eins. Die Übersetzung des C-Codes verlief in erster Linie reibungslos. Es musst aber vorher die Annahme getroffen werden, dass in Zeile drei das „r“ für Radius steht. Hinsichtlich der Optimierung wurde auch hier anstatt der Multiplikation, mit einer Binären Verschiebung nach links gearbeitet.

Zudem konnten einige Operationen bei den Parametern der Funktion „draw\_pixel“ gespart werden, da die richtigen Werte schon in dem richtigen Register stehen.

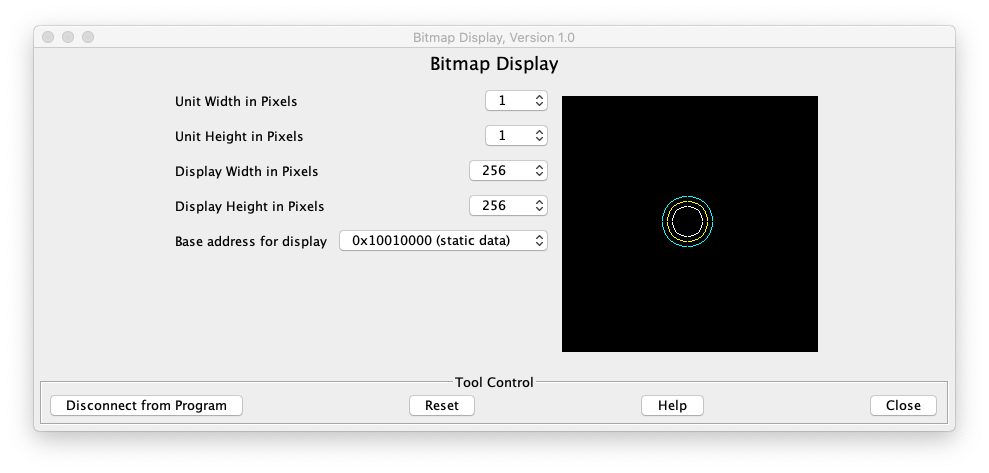
Bei den Tests hatten wir öfters das Problem, dass der Test zwischen positiv und negativ hin und her gependelt ist. Durch einen einfachen Neustart des Programms RARS, waren dann alle Tests positiv, obwohl vor dem Neustart doch ein bis zwei Negativ waren. Dadurch waren wir gezwungen einen großen Teil des Programmes neu zuschreiben, da wir nicht wussten ob der Fehler selbstverschuldet war oder nicht. Nachdem wir RARS auch geupdatet hatten und das Programm neu implementiert hatten, erlangten wir stabile positive Tests.

Abbildung Kreise zeichnen mit Integer-Algorithmus

Uns ist aufgefallen auch mit Rücksprache anderer Gruppen, dass der Kreis an den Positionen ganz oben, rechts, links und unten Lücken besitzt. Wir sind im Folgenden davon ausgegangen, dass unser Code richtig ist, da alle Tests positiv ausgefallen sind und andere Gruppen die gleichen Lücken haben. Wir haben uns trotzdem auf die Fehlersuche gemacht und uns ist aufgefallen, dass wenn der Wert von „y“ nicht gleich Null, sondern gleich „-1“ gesetzt wird, werden die Lücke gefüllt und es entsteht ein vollständiger Kreis (siehe Abbildung 3), welcher auch alle Tests positiv abschließt.

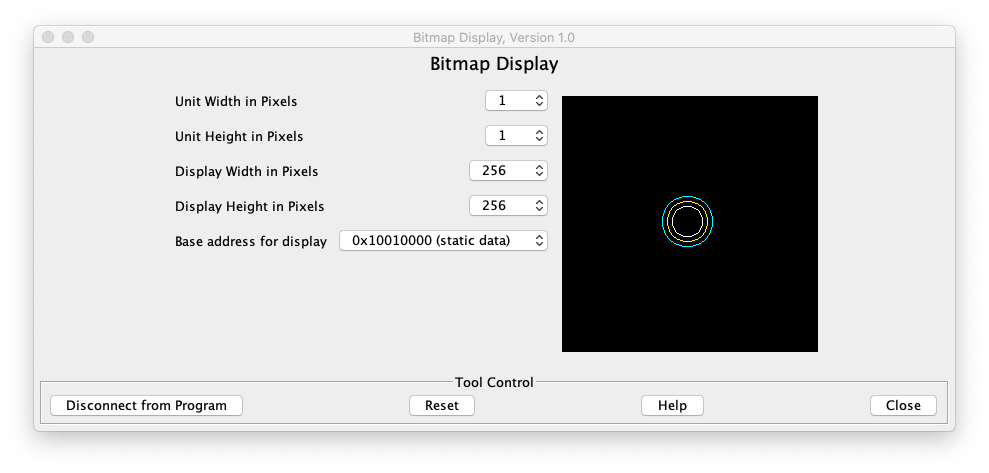


Abbildung Vollständigen Kreis zeichnen mit Integer-Algorithmus

In unserer „circle\_int.asm“ ist die Zeile enthalten, welche zum vollständigen Kreis führt, aber ist auskommentiert.

# Aufgabe 3. Auswertung

Der erste Teil der Aufgabe 3 verlief sehr schnell, da wir nur den Wert ablesen mussten. Der zweite Teil der Aufgabe warf die Frage in den Raum, welche replacement policy wir verwenden müssen. Wir haben uns für die Fully Associative policy entschieden, da wir dort die besten Ergebnisse erhalten haben. Die restlichen Parameter wurden so angepasst das wir die gewünschten 512 Byte Cache Size erreichen. Des Weiteren haben wir beide Programme einen Kreis zeichnen lassen, um sie miteinander zu vergleichen.