Tabelle 2.1: Statistikwerte zu den Cooccurence-Matrizen

	Kachelmuster	Zufallsbild
Energie	608	232
Entropie	188	109
Kontrast	8	127
Homogenität	52	30

In der Cooccurence-Matrix des Zufallsbildes ist die Diagonale nicht mehr verstärkt ausgeprägt, da alle Nachbarschaften etwa gleich häufig vertreten sind. Tabelle 2.1 zeigt die Statistikwerte für die Cooccurence-Matrizen beider Bilder. Die Energie des Kachelbildes ist besonders hoch, da sich die gesamte Anzahl von Benachbarungen auf nur wenige Einträge der Matrix verteilt. Wegen $a^2+b^2 \leq (a+b)^2$ ist die Energie der Cooccurence-Matrix eines homogenen Bildes am höchsten und die Energie eines Bildes mit konstanter Benachbarungshäufigkeit am geringsten. Der Kontrast des Kachelmuster-Bildes ist sehr gering, da meist identische Grauwerte zueinander benachbart sind. Deshalb ist der Wert für die Homogenität deutlich höher als beim Zufallsbild.
Man sieht, daß eine Unterscheidung zwischen einem Zufallsbild und einem stark strukturierten Bild wie dem Kachelmuster anhand der Größen Energie und Kontrast gut möglich ist. Dagegen sind die Statistikwerte zum Histogramm beider Bilder nahezu identisch, da beide Bilder etwa gleiche Histogramme haben.

2.7 Kachelsummen

Unter der Bildung von Kachelsummen (bin accumulation [10], intensity averaging [21]) versteht man die Zerlegung eines Bildes in meist gleichgroße rechteckige und einander nicht überlappende Kacheln, innerhalb derer alle Grauwerte des Ursprungsbildes aufsummiert werden. Die so entstehende Ergebnismatrix entspricht einem mit reduzierter Pixelauflösung (aber höherer Grauwertauflösung) aufgenommenen Bild. Dieser Effekt ist beispielsweise aus dem Fernsehen bekannt, wo er verwendet wird, um die Gesichter von Personen unkenntlich zu machen.

Kachelsumme für die Kachel K mit den Koordinaten $(a*A,b*B)$ im Ursprungsbild $g(x,y)$ bei Kachelgröße $A*B$:

$$K(a,b) = \sum_{x=a\cdot A}^{a\cdot(A+1)-1} \sum_{y=b\cdot B}^{b\cdot(B+1)-1} g(x,y)$$

Die Kachelsumme ist ein einfaches Beispiel einer mit linearen Operatoren durchführbaren Operation. Hier wird allerdings eine starke Reduktion der Pixelanzahl vorgenommen, da alle Kacheln nichtüberlappend angeordnet werden.
Abbildung 2.9 zeigt ein Beispiel für die Bildung von Kachelsummen.

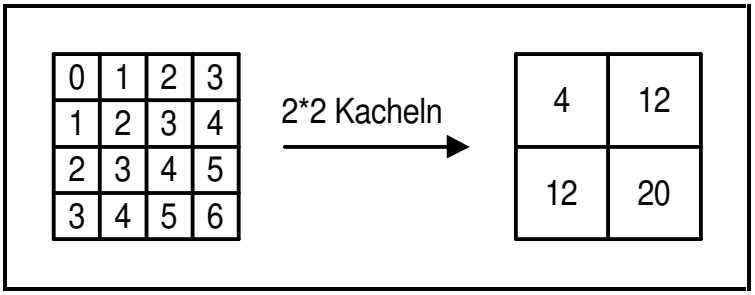

Abbildung 2.9: Bildung von Kachelsummen

Die Zerlegung eines Bildes in Kacheln kann in der Bildverarbeitung verwendet werden, um detaillierter zu untersuchende Bereiche innerhalb des Ursprungsbildes zu selektieren. Dies ist möglich, wenn sich diese Bereiche kontraststark vom Hintergrund abheben. Dann kann bereits bei Abweichung einer Kachelsumme vom Mittelwert eine Selektion des zu untersuchenden Bildbereiches geschehen. Insbesondere bei industriellen Anwendungen mit hohem Kontrast zwischen Werkstück und Hintergrund können so Bereiche, in denen sich Werkstücke befinden, von solchen Bereichen getrennt werden, die nur den Hintergrund enthalten.
Anhand von über ein Bild des Hintergrundes gebildeten Kachelsummen kann die Helligkeitsverteilung innerhalb des aktuellen Bildausschnittes grob untersucht werden, um beispielsweise einen Defekt der Beleuchtung zu erkennen oder anhand des Ergebnisses Helligkeits-Korrekturfaktoren für die Bildauswertung innerhalb der einzelnen Segmente zu berechnen. Dieses Vorgehen ist insbesondere für Binarisierung interessant, wenn das Minimum zwischen den zu trennenden Bereichen nicht stark genug ausgeprägt ist (vgl. Abbildung 2.3: Schwellwertwahl über Histogramm).
Durch den Vergleich zweier über Kachelsummen reduzierter Bilder kann schnell festgestellt werden, wo Unterschiede zwischen den Bildern bestehen, die beispielsweise durch Verschiebung von Gegenständen entstanden sind.

3 Anforderungskatalog und Lösungsansätze

Die Entscheidung, den Histogrammer als ASIC zu implementieren, war bereits vor Beginn der Arbeit gefallen. Bevor ein neuer Chip erstellt wird, ist es jedoch sinnvoll, zu untersuchen, ob die Funktionalität nicht auch mit Software zufriedenstellend nachgebildet werden kann. Deshalb soll untersucht werden, wie aufwendig die Histogrammerstellung als Grundfunktion eines Histogrammers in Software ist.
Im Anschluß daran folgt ein Anforderungskatalog, der die zu implementierende Funktionalität umreißt, aber wesentliche Designentscheidungen bereits nach Abschätzung des Realisierungsaufwandes festlegt.

3.1 Untersuchung einer Softwarelösung

Die Aufzeichnung eines Histogramms ist auf den ersten Blick eine einfache (aber datenintensive) Operation. Es steht außer Frage, daß eine Softwarelösung möglich ist. Sollen nur wenige Histogramme gebildet werden oder ist die Geschwindigkeit der Histogrammerstellung unkritisch, so ist eine Softwarelösung anzuraten. Sollen jedoch bewegte Bilder untersucht werden, oder findet ein Einsatz unter Echtzeitbedingungen statt, wie bei einem industriellen Produktionsprozeß, wird der Zeitfaktor eine kritische Größe. Deshalb ist abzuschätzen, wieviel Zeit ein Standardmikroprozessor zur Histogrammerstellung brauchen würde.
Die Frage nach einer Softwarelösung ist insbesondere deshalb berechtigt, da in den meisten Fällen ein Mikroprozessor zur Auswertung des Histogramms und der Bilddaten verwendet wird.

Entscheidend für die Verarbeitungszeit ist der für die Verarbeitung jedes Pixels notwendige Programmaufwand. Ein Bild von der Kamera soll über eine Framegrabber-Karte auf Anforderung des Rechners eingelesen und gespeichert werden können. Der Rechner kann dann auf die Bilddaten in einem Speicherbereich (Speicher der Framegrabber-Karte) zugreifen.

Ein Programm zur Aufzeichnung des Histogramms muß in seiner Hauptschleife drei Schritte ausführen:
1. Laden eines Helligkeitswertes indiziert über einen Pixel-Adreßzähler
2. Die durch den Helligkeitswert indizierte Histogrammspeicherstelle erhöhen
3. Zähler weiterzählen, auf Endwert prüfen und zurück zu 1., falls noch nicht fertig

Dies kann ein 32-Bit Standardmikroprozessor mit drei Befehlen ausführen. Die üblicherweise 256 oder 1024 Einträge große Histogrammtabelle paßt bei gängigen Prozessoren in den Prozessorcache, so daß der Read-Modify-Write-Zugriff bei Schritt 2 schnell ausgeführt werden kann. Mit einem 32-Bit Zugriff bei Schritt 1 können zwei bis vier Punkte auf einmal geholt werden, so daß Schritt 1 und 3 entsprechend weniger oft ausgeführt werden müssen. In einem Schleifendurchlauf können also mehrere Pixel bearbeitet werden (sogenanntes Loop-enrolling).

Es ist wünschenswert, separate Histogramme über mehrere Fenster innerhalb eines Bildes erstellen zu können.

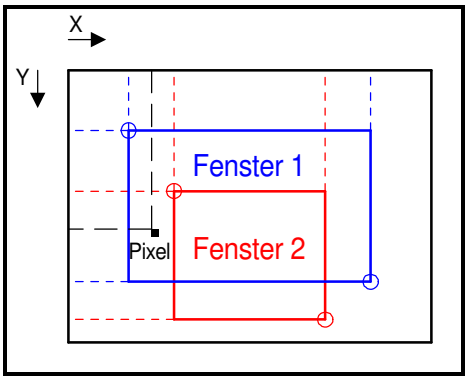

Abbildung 3.1: Aufteilung in Fenster

Zur Realisierung mehrerer Fenster parallel muß für jeden Bildpunkt geprüft werden, in welchen Fenstern er liegt. Dazu ist jedoch die Kenntnis von X- und Y-Wert notwendig, d.h. der Pixeladreßzähler muß in zwei entsprechende Zähler zerlegt werden und die Schleife in zwei verschachtelte Schleifen umgewandelt werden. Nun kann bei Schritt 2 für jedes Pixel ein Koordinatenvergleich mit den Eckpunkten aller Fenster stattfinden (Abbildung 3.1). Vergleiche für Fenster würden jedoch den Programmaufwand in der Schleife und damit die Ausführungszeit vervielfachen.
Eine schnellere Methode zur Implementation mehrerer Fenster ist die fensterweise Aufzeichnung des Histogramms. Abgesehen von der Einschachtelung der oben abgebildeten Schleife für die X-Koordinate in eine Schleife für die Y-Koordinate bliebe der zeitkritische Teil (Schritte 1-3) gleich. Gegenseitige Inklusionen und Überschneidungen von Fenstern könnten durch Aufsummierung der einzelnen Histogramme ausgenutzt werden, so daß jeder Pixel im Bildspeicher insgesamt auch nur einmal abgefragt werden muß.
Weitere Operationen (Minima/Maxima- Bestimmung, Helligkeitssummen, usw.) können anhand der gebildeten Histogramme vergleichsweise schnell ausgeführt werden. Eine Hardwaremöglichkeit zur Egalisierung besteht oft bereits auf der Framegrabber-Karte oder der Grafikkarte in Form eines vom Mikroprozessor ladbaren RAMs, das als Look-Up-Tabelle direkt im Pixelstrom die Egalisierung durchführen kann.

3.1.1 Abschätzung der Ausführungsgeschwindigkeit

Um den Zeitaufwand für die Histogrammerstellung durch den Mikroprozessor abzuschätzen, sei hier ein geeigneter Assembler-Abschnitt für einen 80386-kompatiblen Prozessor bei einer Grauwertauflösung von 8 Bit angeführt:

```
;Hauptschleife:
;EDI enthält die Startadresse der 256*32 Bit großen Histogrammtabelle,
;der Framegrabber ist memory-mapped im Segment ES.
;EDX zeigt auf den Start der zu bearbeitenden Zeile.
;ESI enthält die (durch 4 teilbare) Anzahl der zu bearbeitenden Pixel.
;(Beim 80386 kann auf ein Register folgendermaßen zugegriffen werden:
; Bsp.: AX-Register: EAX=Bits 0..31, davon AX=Bits 0..15, davon
;                    AL=Bits 0..7, AH=Bits 8..15)

            xor EBX,EBX             ;EBX als Tabellenzeiger zu 0 löschen
XSchleife:  mov EAX,ES:[EDX+ESI]    ;4 8-Bit Pixel von Framegrabber holen
            mov BL,AL               ;Unteres Pixel (0) als Zeiger
            inc [EDI+EBX*4]         ;Histogrammspeicherstelle erhöhen
            mov BL,AH               ;nächstes Pixel (1) als Zeiger
            inc [EDI+EBX*4]         ;Histogrammspeicherstelle erhöhen
            ror EAX,16              ;obere 16 Bit bytew. zugreifbar machen
            mov BL,AL               ;nächstes Pixel (2) als Zeiger
            inc [EDI+EBX*4]         ;Histogrammspeicherstelle erhöhen
            mov BL,AH               ;oberes Pixel (3) als Zeiger
            inc [EDI+EBX*4]         ;Histogrammspeicherstelle erhöhen
            sub ESI,4               ;4 Pixel weiter (rückwärts)
            jnc XSchleife           ;kein Unterlauf?, dann weiter
```

Listing 3.1: Histogrammerstellung in Assembler

Die gesamte Taktzyklenanzahl für einen Schleifendurchlauf ergibt sich beispielsweise für einen Intel 486-Prozessor [12] zu:

- Laden der vier Pixel: 7 Takte = 3 Takte + 4 Waitstates des Framegrabbers (Annahme: 4 Waitstates ≈ 70 ns Zugriffszeit)
- Erhöhen der Histogrammeinträge: 18 Takte = 4 * (1 + 3) + 2
- Schleifenabschluß: 4 Takte = 1 + 3

Insgesamt ergeben sich 29 Taktzyklen für 4 Pixel, also ca. 8 Takte pro Pixel. Bei einer Prozessor-Taktfrequenz von 66,7 MHz sind dies also 8*15ns=120 ns pro Pixel.

Da das Bild bereits im Speicher vorliegt, ist eine Verarbeitung mit Bildfrequenz nicht unbedingt notwendig. Angenommen, die Histogrammerstellung dürfte den Prozessor zu 25% auslasten, könnten bei einer Auflösung von 1024*1024 Pixeln noch zwei Vollbild-Histogramme pro Sekunde erstellt werden. Die Erstellung jedes Histogramms dauert dabei 125 ms.

Der ermittelte Wert von zwei nebenläufig erstellten Histogrammen pro Sekunde liegt deutlich unterhalb der benötigten Bildverarbeitungsrate für viele industrielle Anwendungen und weit unterhalb der Bildrate gängiger CCD-Sensoren (ca. 25 bis 100 /s). Eine Wartezeit von über 100 ms auf ein Histogramm würde die Reaktionszeit des Systems stark erhöhen und während der Histogrammerstellung blieben mehrere Bilder ungenutzt.
Durch Einsatz eines schnelleren Prozessors kann die Verarbeitungsgeschwindigkeit um den Faktor vier oder mehr erhöht werden. Die Preisdifferenz wäre allerdings größer als die Kosten einer kommerziell verfügbaren Spezialhardware in Form eines Histogrammchips.
Für niedrige Auflösungen oder geringe Anforderungen an die Reaktionszeit scheint eine Softwarelösung jedoch angemessen zu sein.

3.2 Wesentliche Anforderungen an den Histogrammer

Nach Entscheidung für eine Hardwarelösung muß untersucht werden, wie die Grundfunktion mit ausreichender Flexibilität realisiert werden kann und welche Funktionen neben der eigentlichen Grundfunktion sinnvoll implementiert werden können. Bei diesen Überlegungen ist zwischen mehreren Kriterien abzuwägen:

- Eine größere Flexibilität macht das Produkt langlebiger, erhöht aber die Chipfläche und damit die Kosten.
- Ein größerer Funktionsumfang kann den Hostrechner entlasten. Es ist jedoch zwischen zusätzlichem Aufwand und im Hostrechner eingesparter Rechenzeit abzuwägen.
- Die Erstellung eines Chips ist aufgrund des hohen Entwicklungsaufwandes und der hohen Kosten für Prototypen teuer. Eine Notwendigkeit späterer Modifikationen aufgrund von unzureichendem Funktionsumfang sollte daher vermieden werden.

3.2.1 Grundsätzliche Designentscheidungen

Bevor die Funktionalität festgelegt wird, sind Entscheidungen über die Rahmenbedingungen des Einsatzes zu treffen. Für den Histogrammer wesentliche Parameter sind Bildauflösung und Grauwertauflösung. In den meisten einfachen Bildverarbeitungssystemen wird mit einer Auflösung von 512*512 oder 1024*1024 bei 8 Bit Grauwertauflösung gearbeitet. Es gibt jedoch auch CCD-Sensoren mit bis zu 4096 Pixeln pro Zeile. Insbesondere ist ein Trend zu höheren Pixelanzahlen festzustellen. Grauwertauflösungen von 10 Bit oder mehr werden in speziellen Anwendungen benötigt. Übliche Pixelfrequenzen liegen im Bereich von 10 bis 25 MHz bei Bildfrequenzen zwischen 25 und 100 Hz.
Um den aktuellen Anforderungen des Marktes gerecht zu werden, sollten also zumindest 10 Bit Grauwertauflösung und 1024*1024 Pixel Bildauflösung abgedeckt werden. Um den Anforderungen innerhalb der Nutzungsdauer des Histogrammers ebenfalls gerecht zu werden, sollten insbesondere auch höhere Auflösungen vorgesehen werden. Obwohl bei höherer Auflösung in vielen Anwendungen auch höhere Pixelfrequenzen notwendig werden, ist es nicht unbedingt notwendig, bereits die erste Version des Histogrammers dafür ausreichend schnell zu machen. Dies kann auch später durch Übergang auf kleinere Chip-Geometrien geschehen.

Anhand einer Liste möglicher Aufgaben eines Histogrammers [23] wurden Vorüberlegungen über die zu implementierende Funktionalität gemacht. Anhand dessen wurde [24] am MAZ als Pflichtenheft erstellt. Neben den Anforderungen an den Funktionsumfang ist ein Basiskonzept des Histogrammers mit dem Vorschlag einer Teilspezifikation enthalten. Die vorgeschlagene Spezifikation wurde in wesentlichen Punkten nicht übernommen (s. Kapitel 4.5), um eine einfache Einbindung des Histogrammers in Rechnersysteme zu ermöglichen. Deshalb wird [24] im folgenden als Pflichtenheft bezeichnet. Abweichungen von diesem Pflichtenheft in Form von Abänderung oder Erweiterung vorgesehener Funktionen im Histogrammer oder Auslagerung von Funktionen auf den Hostrechner erwiesen sich im Verlauf der Arbeit als sinnvoll und wurden in Kooperation mit dem MAZ diskutiert und vereinbart.
Im folgenden sollen die wesentlichen Grundzüge aus dem Pflichtenheft des MAZ [24] hinsichtlich der Implementation ausgeführt und erläutert werden. Dabei wird auf diejenigen Punkte des Pflichtenheftes, in denen das Konzept noch grundlegend geändert wurde, nicht oder nur kurz eingegangen.

Der Histogrammer soll mit Bildauflösungen von bis zu 4096*4096 Pixeln und Grauwertauflösungen von bis zu 10 Bit arbeiten. Die erreichbare Pixelfrequenz wird im Pflichtenheft nicht vorgegeben. Sie wird vom Design und der Strukturgröße bestimmt werden. Vorerst

werden 20 MHz als Minimum bei einer Realisierung in 0,8 µm-Technologie angestrebt. Eine Pixelfrequenz von 20 MHz wird von vielen heutigen Bildverarbeitungssystemen unterstützt und wird ebenfalls von Konkurrenzprodukten erreicht.

3.2.2 Histogrammerstellung

Das Konzept des Histogrammers ähnelt den kommerziell erhältlichen Histogrammern von LSI Logic [21] und Harris [10]. Beide Bausteine bestehen in ihrem Kern aus einem schnellen Speicher, der zur Abspeicherung der Histogramme verwendet wird. Alle Operationen lesen ihre Argumente aus dem Speicher und schreiben das Ergebnis in diesen zurück. Anhand der wichtigsten Funktion des Histogrammers, der Histogrammerstellung, ist festgelegt worden, wie groß der Histogrammspeicher sein soll. Die weiteren Funktionen sind an das Speicherangebot angepaßt definiert worden.

Der Histogrammprozessor soll bei den unterschiedlichen Auflösungen von 512*512 bis 4096*4096 Pixeln Histogramme über das gesamte Bild bilden können. Bei einer Grauwertauflösung von 10 Bit ist also eine Anzahl von 1024 Speicherstellen für je einen Wert des Histogramms notwendig. Jede dieser Speicherstellen muß Werte von bis zu 4096*4096= 2^24= 16,78 Millionen aufnehmen können, da im Extremfall alle Punkte des Bildes den gleichen Grauwert haben und die entsprechende Histogrammspeicherstelle dann von 0 ausgehend 2^24-mal inkrementiert wird.

Der Speicherbedarf für die Akkumulation der Grauwerte eines Histogramms bei Auflösung $n*m$ ist:
$G \cdot \log_2(n \cdot m)$ Bit.
Haben alle Pixel innerhalb des Bildes den gleichen Grauwert, tritt dann allerdings ein Überlauf auf 0 in der betroffenen Speicherstelle auf, da die maximal darstellbare Zahl $n*m$-1 ist.

Um gleichzeitig zusätzlich zum Histogramm des Vollbildes Histogramme über ein beliebiges Fenster innerhalb des Bildes aufzeichnen zu können, soll der doppelte Speicherplatz, also 1024*48 Bit, vorgesehen werden.

Da vorauszusehen war, daß der Speicher einen nicht unwesentlichen Flächenanteil des Chips ausmachen wird, wurde eine Flächenabschätzung mit einem Speichergenerator-Werkzeug vorgenommen. Der geplante 6 kB große Speicher belegt ca. 25 Quadratmillimeter Chipfläche bei Realisierung mit einem 1,0 µm CMOS-Prozeß. Bei einer Fertigung in einem Prozeß mit feineren Strukturbreiten verringert sich die Fläche entsprechend dem Quotienten der Quadrate der Strukturbreiten. Bei 0,8 µm-Technologie belegt der Speicher also eine Fläche von etwa
25 mm² * 0,8² / 1,0² = 16 mm².
Der Wert von 25 mm² bei 1,0 µm-Technologie erscheint akzeptabel. Eine Vervielfachung, um höhere Grauwertauflösungen oder eine größere Anzahl von frei programmierbaren Fenstern zur Verfügung zu stellen, würde den Chip jedoch schnell zu groß werden lassen. Eine Chipfläche von insgesamt etwa 100 mm² sollte bei einem derartigen Entwurf keinesfalls überschritten werden. Die Möglichkeit, einen externen Speicher zur Akkumulation der Histogramme zu verwenden, wurde aufgrund der hohen Pinanzahl für Datenleitungen und der unvermeidbaren Geschwindigkeitseinbußen verworfen.

Bei niedrigerer Auflösung oder kleineren Fenstern ist es mit dem gleichen Speicheraufwand auch möglich, parallel mehrere Fenster aufzuzeichnen. Vier frei positionierbare Fenster sollen vorgesehen werden. Werden beispielsweise alle vier Fenster gleichmäßig auf die Speicherwortbreite von 48 Bit aufgeteilt, ergibt sich also eine Speicherbreite von 12 Bit pro Fenster,

entsprechend einer Fenstergröße von bis zu 64*64 Pixeln (s. Abbildung 3.2: Aufteilung des Histogrammspeichers). Da eine frei programmierbare Zuteilung des 48 Bit breiten Speicherwortes auf die vier Fenster einen ungerechtfertigt hohen Hardwareaufwand verursachen würde, ist eine Anzahl sinnvoller Zuteilungen festgelegt worden (s. [24] und Kapitel 4.5.2). Für jede Bildgröße mit einer Kantenlänge in X- und Y- Richtung von bis zu 2^a Pixeln, mit a∈ {8,9,10,11,12}, wurde mindestens eine Speicherzuteilung definiert, so daß der nicht für das Histogramm über das Vollbild benötigte Speicherplatz für Histogramme über Teilfenster zur Verfügung steht. Diese Zuteilung ist unabhängig von der genutzen Grauwertauflösung.

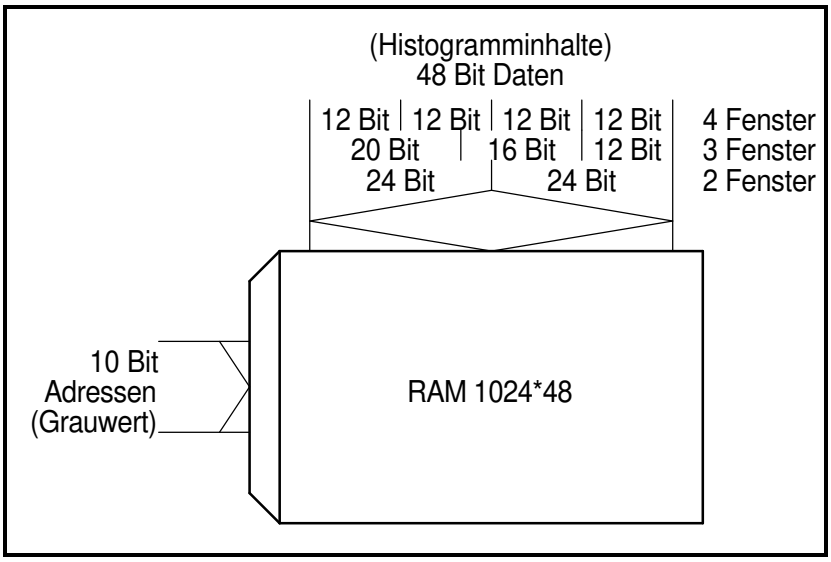

Abbildung 3.2: Aufteilung des Histogrammspeichers

Denkbar wäre jedoch auch eine Teilung des gesamten Speicherarrays in zwei oder mehr Bereiche, so daß bei geringerer Graustufenanzahl eine höhere Speicherbreite für jedes Fenster zur Verfügung steht. Beispielsweise könnten mit der gewählten Speicherkapazität noch bei 9 Bit Grauwertauflösung vier Histogramme mit bis zu 4096*4096 Pixeln aufgezeichnet werden.

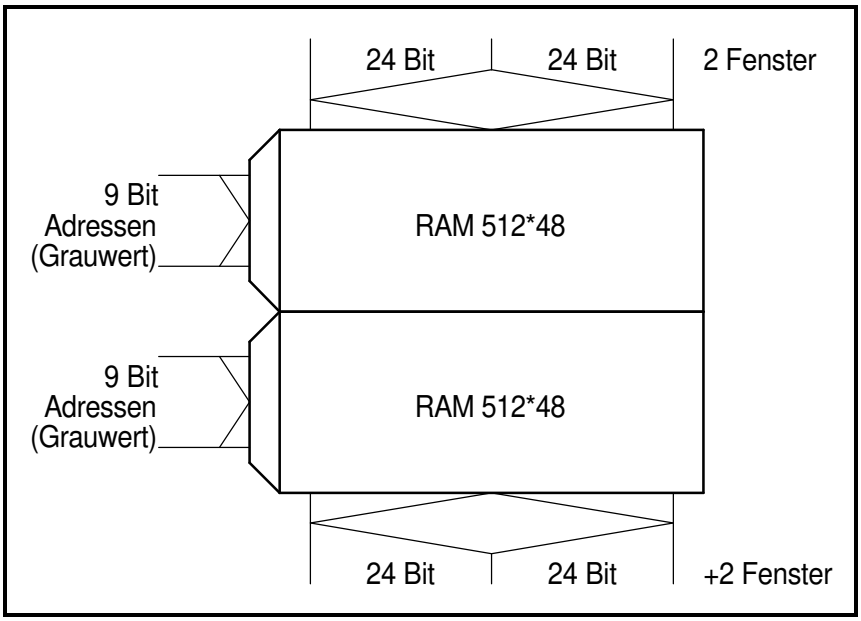

Abbildung 3.3: Alternative Aufteilung des Histogrammspeichers

Diese Option wurde nicht vorgesehen.

Neben der eigentlichen Histogrammerstellung sind weitere Funktionen zum Löschen des Histogrammspeichers vor der Histogrammerstellung und zum Auslesen der Histogramme notwendig. Die Löschfunktion soll aus Sicht des Anwenders als *Flash-Clear* realisiert werden, d.h. sie läuft ohne Interaktion des Hostrechners ab, nachdem sie gestartet wurde.

3.2.3 Egalisierungsfunktionen

Um ein Bild egalisieren zu können, muß zunächst ein Histogramm des Bildes erstellt werden, dann eine Umsetzungstabelle erstellt werden und anschließend jedes Pixel des Bildes anhand dieser umgewandelt werden. Eine dazu notwendige Zwischenspeicherung des Bildes auf dem Histogrammer wäre jedoch aufgrund des immensen Speicheraufwandes nicht zu realisieren. Deshalb soll der Histogrammer nach der Aufzeichnung eines Histogramms eine Egalisierungstabelle für dieses Bild erstellen können, und erst darauf folgende Bilder anhand dieses Histogramms egalisieren können. Da sich der Bildinhalt im allgemeinen von Bild zu Bild nur geringfügig ändert, ist dieses Vorgehen sinnvoll. Der im Histogrammer vorhandene Speicher soll als Look-Up-Tabelle (LUT) zur Umsetzung eines eingehenden Grauwertes in einen neuen Grauwert verwendet werden können. Zu diesem Zweck erhält der Histogrammer einen Pixelausgang für die umgewandelten Grauwerte.
Der in Kapitel 2.4 beschriebene vereinfachte Algorithmus zur Egalisierung (Egalisierung 1), kann einfach im Histogrammer implementiert werden, da er keine Wenn-Dann-Entscheidungen enthält, sondern einem festgelegten Funktionsablauf folgt.
Im Histogrammspeicher wird die Summenhäufigkeitsverteilung gebildet. Dabei wird das Histogramm überschrieben:

$$Hist_{neu}(i) := \sum_{j=0}^{i} Hist_{alt}(j) .$$

Die Pixelsummen müssen durch die Pixelanzahl dividiert und mit der (gewünschten) Grauwertauflösung multipliziert werden. Die dazu je Eintrag notwendige Division kann bei der Bildung der Pixelsummen geschehen und das auf eine ganze Zahl gerundete Ergebnis direkt in den Speicher eingeschrieben werden. Bei dieser Operation findet jedoch ein unwiderrufbarer Informationsverlust statt. Eine nachträgliche Erhöhung der Grauwertauflösung ist dann nicht mehr möglich. Dieser Informationsverlust kann umgangen werden, indem die Division erst direkt beim Auslesen der Egalisierungstabelle für jedes Pixel vorgenommen wird. Bei industriellen Anwendungen kommen oft CCD-Kameras mit gleicher Auflösung in X- und Y-Richtung und Zweierpotenzen als Kantenlänge zum Einsatz [32], so daß der Divisor eine Zweierpotenz ist. Da ein frei einstellbarer Divisor eine sehr aufwendige Divisionsschaltung erfordern würde, soll die Division durch einen Multiplexer als binäre Verschiebung realisiert werden (s. Abbildung 3.4: Verwendung des RAMs als LUT). Der dabei aufgrund des nur auf festgelegte Zweierpotenzen einstellbaren Divisors unter Umständen entstehende Verlust an Grauwertauflösung wird in Kauf genommen. Dieses Verfahren wird auch bei dem kommerziellen Baustein L64250 [21] verwendet.

Da der Speicher des Histogrammers testbar sein muß, ist es sinnvoll, ihn vom Hostrechner her ladbar zu machen. Auf diesem Weg kann dann auch der Hostrechner eine Egalisierungstabelle oder eine beliebige andere Umsetzungstabelle in den Histogrammer laden. Im 48 Bit breiten Speicher ist Platz für bis zu vier 10 Bit breite Tabellen. Durch Implementation geeigneter Stellungen des Multiplexers können alle vier Tabellen abwechselnd genutzt werden.

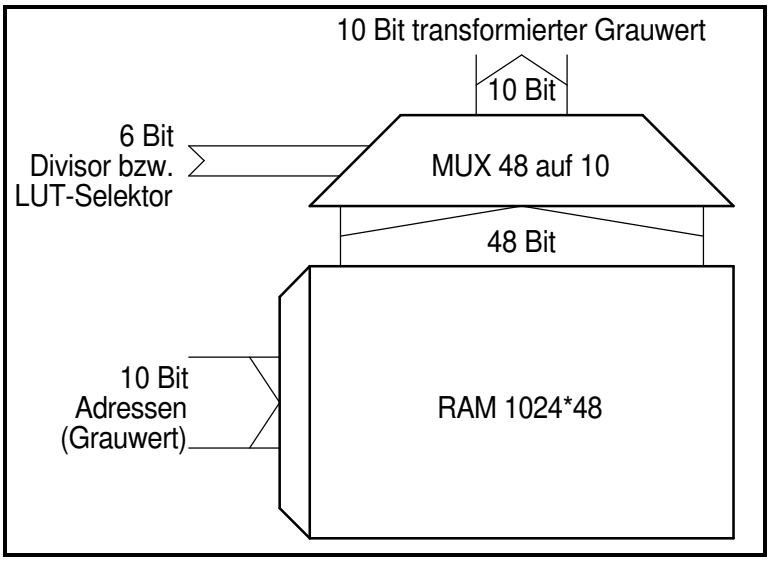

Abbildung 3.4: Verwendung des RAMs als LUT

3.2.4 Cooccurence-Matrix-Erstellung

Die Erstellung einer Cooccurence-Matrix basiert auf der Beachtung einer Nachbarschaftsrelation. Dazu muß bei jedem Pixel des Bildes auf seinen Nachbarn bezüglich der Relation zugegriffen werden können. Hier sollen zur Vereinfachung nur Nachbarschaftsrelationen zwischen jeweils zwei Pixeln betrachtet werden. Die Realisierung komplexerer Nachbarschaftsfunktionen ist durch Addition der zugrundeliegenden Coocurence-Matrizen über Zweiernachbarschaften möglich.
Um die Nachbarschaft zwischen Punkten in benachbarten Zeilen des Bildes zu untersuchen, ist es also zumindest notwendig, bei der Untersuchung jeder Zeile die zugehörigen Pixel der jeweils vorigen Zeile zu kennen. Da der Histogrammer nicht wahlfrei auf die Bilddaten zugreifen kann, sondern das Bild in Form eines seriellen Datenstroms übertragen bekommt, müßte er eine gesamte Bildzeile zwischenspeichern können, um eine solche Relation verwirklichen zu können. Um den dazu notwendigen Speicherplatz einzusparen, wird die Konvention getroffen, daß die miteinander in Relation stehenden Pixel jeweils gleichzeitig extern an den Histogrammer angelegt werden. In diesem Fall ist also vor den Histogrammer eine Logik zu schalten, die die Pixeldaten in geeigneter Form liefert.

Mit der im Histogrammer zur Verfügung stehenden Anzahl von 1024 Speicherstellen kann eine Cooccurence-Matrix der Größe 32*32 erstellt werden. Dies entspricht einer Grauwertauflösung von 5 Bit. Damit können also über den existierenden 10 Bit breiten Pixeleingang jeweils zwei zueinander in der Nachbarschaftsrelation stehende Pixel zugleich übergeben werden (Abbildung 3.5: RAM-Adressierung bei Erstellung von Cooccurence-Matrizen). Die zur Erstellung der Cooccurence-Matrix innerhalb des Histogrammers notwendigen Operationen sind damit zurückgeführt auf die zur Histogrammerstellung benötigten Operationen: Aus den Grauwerten der zugleich übergebenen Pixel a und b wird die Adresse g_a+32*g_b gebildet und die darüber adressierte Speicherstelle wird inkrementiert.
Es ist deshalb auch möglich, das Bild wie bei der Histogrammerstellung in mehrere Fenster aufzuteilen, über die jeweils eine Matrix erzeugt wird.

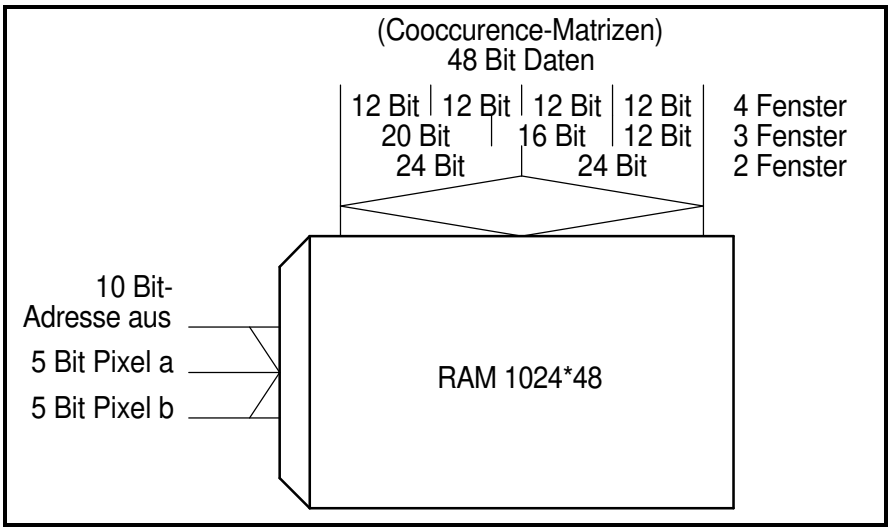

Abbildung 3.5: RAM-Adressierung bei Erstellung von Cooccurence-Matrizen

3.2.5 Statistikfunktionen

Der Histogrammer soll eine Anzahl statistischer Größen liefern, die entweder als Zusatzinformation zum Histogramm genutzt werden können oder in bestimmten Fällen sogar als Entscheidungskriterium dafür verwendet werden können, ob das Histogramm überhaupt ausgewertet werden soll. In Kapitel 2.6 wurde gezeigt, welche Statistikgrößen aus dem Histogramm bzw. der Cooccurence-Matrix gebildet werden können. Da die Errechnung der Statistikgrößen aus einem Histogramm bzw. einer Cooccurence-Matrix aufgrund der relativ geringen Datenmenge an die Leistungsfähigkeit des Hostrechners keine großen Anforderungen stellt, sollen nur die wichtigsten Größen oder Hilfsgrößen direkt vom Histogrammer berechnet werden. Auf Funktionen mit hohem Hardware-Realisierungsaufwand wie z.B. Fließkommarechnungen oder Divisionen soll soweit möglich verzichtet werden.

Eine grundlegende Funktion für viele der Statistikgrößen ist der Mittelwert (s. Kapitel 2.6.1). Zusammen mit dem quadratischen Mittelwert kann die Varianz ermittelt werden. Die Berechnung dieser Größen erfordert Addition, Quadrierung und Division. Da die Division jedoch eine in Hardware aufwendig zu realisierende Funktion ist und nur insgesamt eine Division pro Statistikwert benötigt wird, soll der Histogrammer auf die Division durch die Pixelanzahl verzichten. Diese Funktion kann der Hostrechner ohne wesentlichen Zeitaufwand übernehmen. Die Berechnung innerhalb des Histogrammers reduziert sich auf die Bildung der Grauwertsumme bzw. der Quadratsumme.
Wenn dem Hostrechner minimaler und maximaler Grauwert innerhalb des Histogramms bekannt sind, kann die Anzahl der zu untersuchenden Histogrammeinträge eingeschränkt werden und es wird Rechenzeit eingespart.
Die über einen Zähler zu bildende Schwellensumme hilft dem Hostrechner zu überprüfen, ob die für eine Binarisierung zugrundegelegten Annahmen über die Flächenanteile innerhalb des Bildes noch zutreffen (vgl. Kapitel 2.3).

Deshalb soll der Histogrammer die fünf Größen Minimum, Maximum, Schwellensumme, Grauwertsumme und Quadratsumme über jedes der vier einstellbaren Fenster bilden.

Diese Größen können wahlweise aus dem Histogramm oder direkt aus den Pixeldaten errechnet werden.

Für die Berechnung aus dem Histogramm spricht, daß hier keine hohen Anforderungen an die Verarbeitungsgeschwindigkeit der Addierer, Quadrierer und Komparatoren gestellt werden, da nur 1024 Einträge zu untersuchen sind. Nachteilig ist jedoch, daß die Histogrammeinträge bei der Bildung der Summe und Quadratsumme mit dem zugehörigen Grauwert multipliziert werden müssen und so Argumente sehr hoher Breite entstehen. Weiterhin würden die unterschiedlichen Aufteilungen des Histogrammspeichers Multiplexer erfordern, um auf die Histogramme über die einzelnen Fenster zugreifen zu können.
Eine direkte Berechnung der Statistikdaten aus den Pixeldaten hat dagegen den Vorteil, daß die zu beachtenden Argumente, also die Grauwerte, eine Breite von nur 10 Bit haben und daß die Berechnung auch dann stattfinden kann, wenn kein Histogramm gebildet wird, also beispielsweise bei Benutzung des Histogrammers als Look-Up-Tabelle (s. Kapitel 3.2.3).
Da insbesondere die schnelle Verfügbarkeit der Statistikwerte bei allen Operationen von Vorteil ist, sollen die genannten Größen direkt aus den Pixeldaten errechnet werden.

3.2.6 Kachelsummen-Erstellung

Der Histogrammer soll über das gesamte Bild Kachelsummen erstellen können. Zur Erstellung der Kachelsummen soll der vorhandene Speicherplatz verwendet werden. Da dies eine gleichzeitige Histogrammerstellung ausschließt, muß der Chip zur Erstellung der Kachelsummen in einen anderen Modus umgeschaltet werden, der Kachelmodus genannt werden soll.
Mit der verfügbaren Anzahl von 1024 Speicherstellen läßt sich das Bild in 32*32=1024 Bereiche zerlegen. Bei der maximalen Auflösung von 4096*4096 Pixeln ist eine Kachel also 128*128=2^7*2^7 Pixel groß. Bei einer Grauwertauflösung von 10 Bit werden zur Darstellung der Summe maximal 7+7+10=24 Bit benötigt. Für die Kantenlänge der Kacheln werden nur Zweierpotenzen vorgesehen, um die zur Adreßerzeugung aus den Koordinaten notwendige Division mit einem Multiplexer als binäre Schiebeoperation realisieren zu können. Da Bildformate mit einem Kantenverhältnis von weniger als 1:2 praktisch nicht vorkommen (es sei denn bei Zeilensensoren), werden nur quadratische Kacheln vorgesehen. Die Abbildung (Abbildung 3.6: Adreßerzeugung im Kachelmodus) zeigt die Erzeugung der RAM-Adresse aus den höchstwertigen genutzten Bits der Bildkoordinaten.

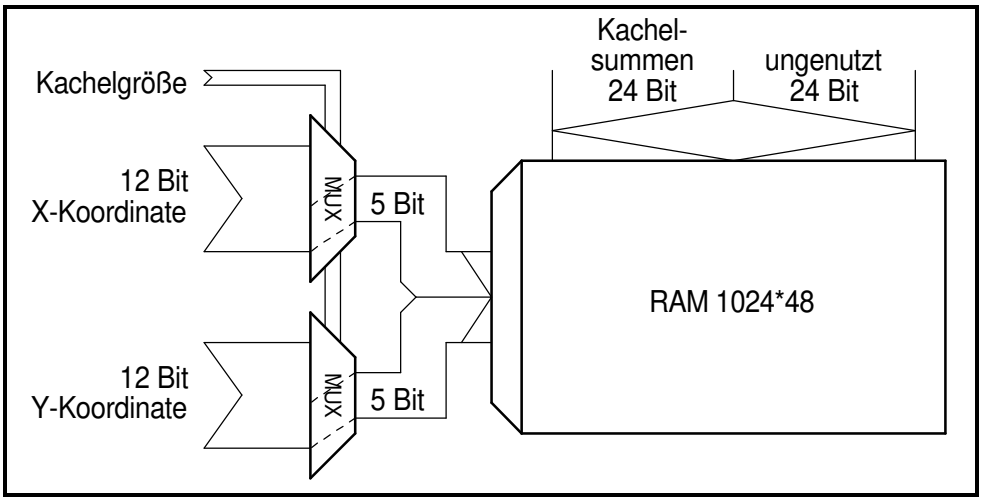

Abbildung 3.6: Adreßerzeugung im Kachelmodus

Eine von der Bildauflösung unabhängige Auswahl der Kachelgröße ermöglicht es, bei Auflösungen unterhalb von 2048*2048 die Anzahl der Kacheln auch geringer als 1024 auszuwählen.

3.2.7 Fensterdefinition, weitere Funktionen

Die vier einstellbaren Fenster, innerhalb derer der Histogrammer Histogramme bildet oder Statistikgrößen errechnet, sollen rechteckig und parallel zu den Bildkanten über Register frei definiert werden können. Dazu muß der Histogrammer die Position jedes Pixels innerhalb des Bildes kennen. Bildhöhe und Breite sollen über Register innerhalb des Histogrammers definiert werden können. Der Histogrammer bekommt über ein Signal vorgegeben, wann ein Bild anfängt. Das Signal soll mindestens bis zum Bildende aktiv sein. Während das Signal aktiv ist werden alle Bildpunkte zeilenweise von links oben nach rechts unten mit je einem Takt übergeben. Intern soll der Histogrammer zählen, an welcher Koordinate sich der aktuelle Bildpunkt befindet. Zusätzlich zu der Möglichkeit, die Bildgröße und die Positionen und Größen der Fenster über Register innerhalb des Histogrammers zu definieren, wird eine Steuerung über Eingänge des Chips vorgesehen. Die Bildgröße soll durch vorzeitiges Signalisieren eines Bildendes über Rücknahme des Signales, das gültige Pixeldaten am Eingang anzeigt, verringert werden können. In welchen der vier Fenster sich ein Pixel befindet, soll über vier parallel zu den Pixeldaten zu liefernde Signale (Window Enable: winEn) bestimmt werden können:

$\text{FensterAktiv}(i) := [X_1(i) \le X \le X_2(i)] \wedge [Y_1(i) \le Y \le Y_2(i)] \vee \text{externe_Steuerung}(i) \wedge \text{winEn}(i)$
i: Nummer des Fensters, $i \in \{1,2,3,4\}$

Die Steuerung der Fensterpositionen über Register geschieht durch die gezeigte Festlegung über die linke obere (X_1,Y_1) und rechte untere Ecke (X_2,Y_2). Alternativ wäre auch die Angabe eines Eckpunktes und der Fenstergröße möglich. Sämtliche Werte können über 12 Bit breite Register definiert werden.

In mehreren Fällen kann innerhalb des Histogrammers ein Überlauf auftreten. Überläufe sollen erkannt und gemeldet werden. Da es jedoch viele mögliche Quellen für Überläufe gibt und ein Großteil der Überläufe durch einfaches Abschneiden des Wertes auf den Maximalwert ohne wesentliche Konsequenz für das Bildverarbeitungssystem verhindert werden kann, wurde in der Realisierung diese Methode bevorzugt.
Ein Überlauf kann bei Histogrammerstellung auftreten, wenn alle Pixel eines maximal großen Fensters den gleichen Grauwert haben, da ein binärer Zähler, wie er aus Speicherstelle und Inkrementer gebildet wird, nach 2^n Inkrementierungen wieder auf Null überläuft. Bei der Bildung der Summenhäufigkeitsverteilung innerhalb des Histogrammspeichers tritt für alle Grauwerte ab dem höchsten im Histogramm vorkommenden Grauwert ein solcher Überlauf ein (s. Kapitel 2.4).

Neben den bisher genannten Funktionen soll der Histogrammer direkt ein Bild binarisieren können. Um wahlweise Pixel mit einem hohen Grauwert auf eine logische 1 abzubilden und Pixel mit einem niedrigen Grauwert auf eine logische 0, oder umgekehrt, wird ein oberer (S_{hi}) und ein unterer Schwellwert (S_{lo}) eingeführt. Beide werden von einem digitalen Fensterkomparator mit dem Grauwert *i* jedes Pixels verglichen. So ist es auch möglich, nur einen bestimmten Grauwertbereich auf eine logische 1 abzubilden. Das Ergebnis dieses Vergleiches wird an einem zusätzlichen Ausgang *Interval* ausgegeben.

$$Interval = \begin{cases} 1 | S_{lo} \le i \le S_{hi} \\ 0 \quad sonst \end{cases}$$

Da die Wahl der Schwellwerte von einer Anzahl von Faktoren abhängt, die dem Histogrammer nicht bekannt sind (vgl. Kapitel 2.3), soll der Schwellwert frei programmierbar sein.

4 Spezifikation und Implementation

Aufgrund des Anforderungskataloges sollen hier zuerst diejenigen Komponenten des Histogrammers untersucht werden, die grundlegende Funktionen haben oder deren Realisierung absehbar kritisch ist. Anhand dessen soll ein Konzept zur Realisierung erstellt werden, um die Realisierbarkeit der Anforderungen aus dem Anforderungskatalog zu überprüfen.
Damit sichergestellt ist, daß der Chip effizient eingesetzt werden kann, wird die endgültige Spezifikation dann aus der Sicht des Programmierers und des Hardwaredesigners eines Systems, das den Histogrammer verwendet, festgelegt werden.

4.1 Übersicht über die zu implementierenden Einheiten

Anhand des Anforderungskataloges kann ein Blockschaltbild erstellt werden, in dem die wesentlichen Einheiten des Histogrammers als „black box“ dargestellt werden (Abbildung 4.1). Man erkennt, daß der Speicher eine zentrale Stellung innerhalb des Histogrammers einnimmt. Die Speicheradresse wird wahlweise aus den Grauwerten oder vom Bildzähler generiert. Die ALU (Arithmetic Logic Unit) nimmt alle internen Modifikationen des Speicherinhaltes vor. Sie enthält Inkrementer zur Erstellung von Histogrammen und einen Addierer zur Erstellung von Kachelsummen. Ein Halteregister (Akkumulator) innerhalb der ALU ermöglicht die Bildung der Summenhäufigkeitsverteilung. Das RAM ist zusammen mit der ALU und der Adreßerzeugung entscheidend für die Leistungsfähigkeit des Histogrammers. Während Businterface und der Pixeleingang und -ausgang grundlegend die Sicht des Histogrammers von außen her bestimmen, prägt die Organisation der Steuerregister wesentlich die Sicht von der Softwareseite.

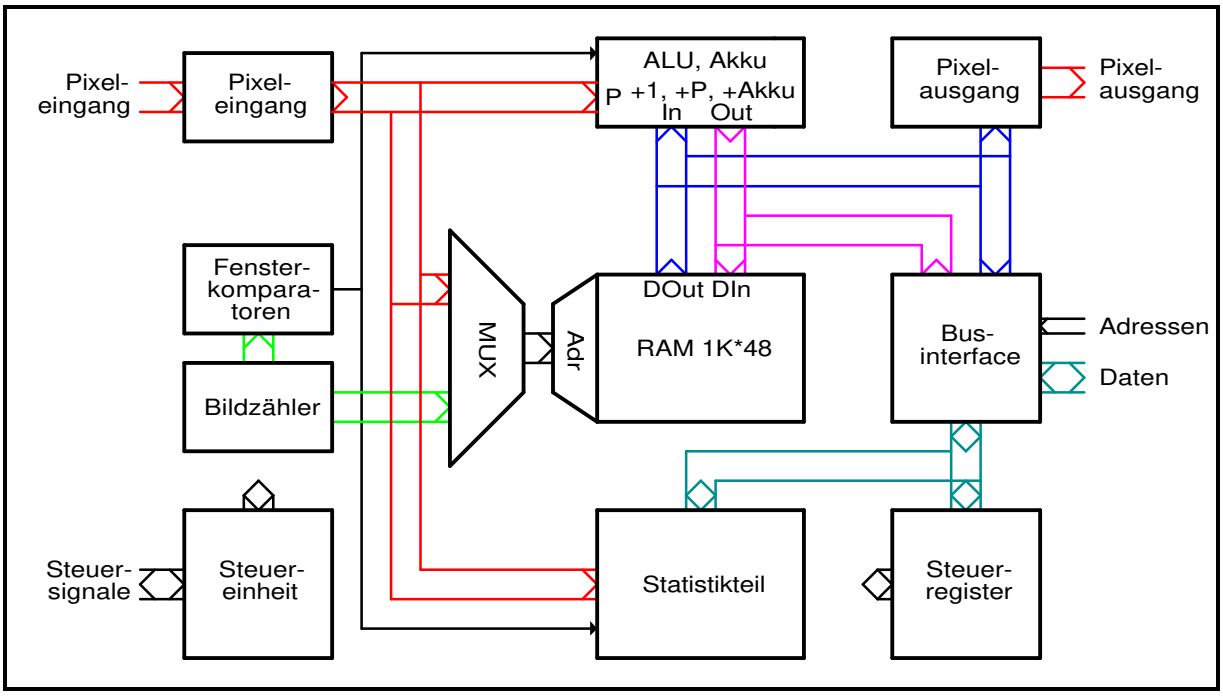

Abbildung 4.1: Blockschaltbild des Histogrammers

Bildpositionszähler und Fensterkomparatoren sollen zuerst untersucht werden, da sie auch einzeln funktionsfähig sind und grundlegende Steuersignale für viele andere Einheiten liefern.

4.2 Positionszähler und Fensterkomparatoren

Der Zähler für die aktuellen Koordinaten innerhalb eines Bildes hat eine Schlüsselfunktion in der Ablaufsteuerung des Histogrammers. Er ist Referenz für die Komparatoren, die bestimmen, wo jedes der vier einstellbaren Fenster aktiv ist, wird bei der Kachelsummenerstellung benötigt, um die Speicheradresse zu erzeugen und wird mit der Endposition verglichen, um das Ende der Bildübertragung festzustellen. Weiterhin kann der Zähler für die Speicheradressierung beim Löschvorgang und bei der Bildung der Summenhäufigkeitsverteilung verwendet werden.
Für die gewünschte Auflösung von bis zu 4096*4096 Pixeln müssen der Zähler für die X- und die Y-Koordinate sowie die Register zur Festlegung der Fensterkoordinaten jeweils 12 Bit breit sein. Es wird das für Bildschirmgeräte übliche Koordinatensystem mit dem Nullpunkt in der linken oberen Ecke gewählt (s. Kapitel 2.2). Die Beschränkung der Zähler und der Fensterkoordinaten auf 12 Bit erfordert, daß die inneren Fenstereckpunkte zu Definition der Fensterkoordinaten verwendet werden.
Um den Hardwareaufwand für den Vergleich der vier Fensterkoordinaten mit den Bildkoordinaten für jedes der vier Fenster zu minimieren, wurde eine Realisierung mit Komparatoren gewählt, die nur auf Gleichheit der aktuellen Koordinaten mit den Eckpunkten prüft. Die Realisierung eines Größenvergleichs zwischen aktueller Koordinate und Start- und Endposition in X- und Y-Richtung ist wesentlich aufwendiger, da ein Größenvergleich je Richtung zwei Subtrahierer erfordert.

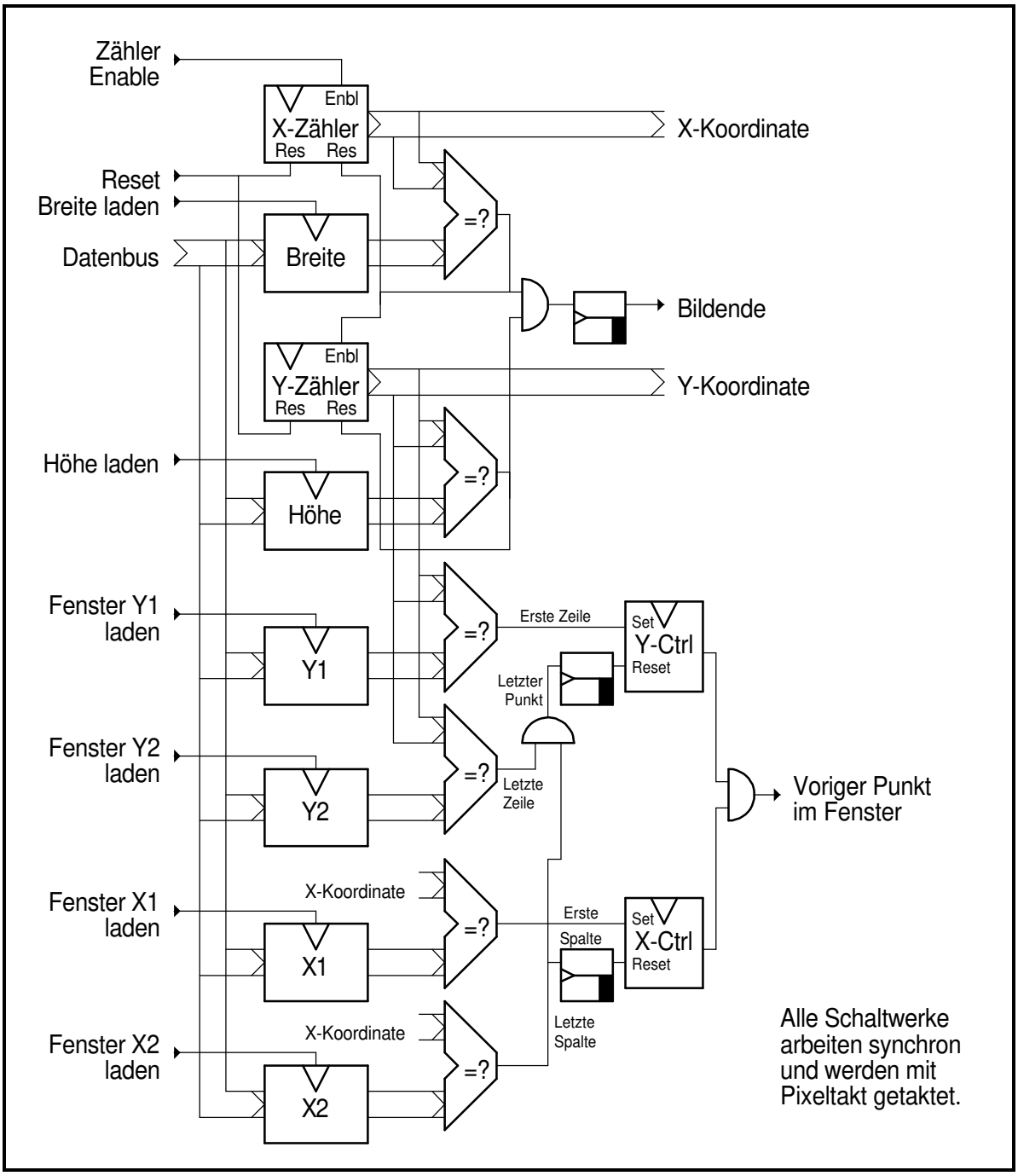

Abbildung 4.2: Bildzähler und Komparator

Abbildung 4.2 zeigt den Bildzähler und die für ein Fenster notwendige Vergleichslogik. Die Fensterkoordinaten werden über die linke obere Ecke (X1,Y1) und rechte untere Ecke (X2,Y2) angegeben. Beim Vergleich der Koordinaten mit den Eckpunkten wird die Information, ob die letzte Spalte (X2) bzw. letzte Zeile (Y2) erreicht wurde, um ein Pixel bzw. eine Zeile verzögert, da der Eckpunkt noch innerhalb des Fensters liegt. Ein Pixel liegt immer dann innerhalb des Fensters, wenn die erste Zeile und die erste Spalte des Fensters bereits erreicht wurde, aber seitdem noch nicht die letzte Zeile oder letzte Spalte überschritten wurde.

4.3 Histogrammteil

4.3.1 Realisierungsmöglichkeiten des Histogrammteils

Die Kernfunktion des Histogrammers ist die Ermittlung der Pixelanzahl zu jeder Helligkeitsstufe. Diese Aufgabe wird vom Histogrammteil wahrgenommen. Neben dem im Pflichtenheft gezeigten Konzept des Histogrammteils sind weitere Realisierungsmöglichkeiten denkbar. Die unterschiedlichen Realisierungsmöglichkeiten sollen im Hinblick auf erreichbare Verarbeitungsgeschwindigkeit und den notwendigen Flächenaufwand auf dem Chip untersucht werden. Das Ziel dabei ist, bei möglichst geringem Flächenaufwand die benötigte oder eine höhere Verarbeitungsgeschwindigkeit (vgl. Kapitel 3.2.1) zu erreichen.

Eine naheliegende Möglichkeit zur Realisierung des Histogrammteils ist die Verwendung jeweils eines Inkrementers (Zählers) pro Helligkeitsstufe für die Speicherung und Erhöhung der Werte. Dieser Ansatz wurde bereits vor 1984 als monolithischer IC in NMOS-Technologie für eine Anzahl von 64 Graustufen bei 18 Bit Positionsauflösung (512*512) realisiert [15].
Begründet wird diese Realisierung folgendermaßen:
Zitat aus [15], Abschnitt 5.4.2:
„The 2K bit (64x18) memory requirement and the 16 MHz video rate dictate the choice of technology [Anm.: NMOS]. However an 18 bit addition cannot be accomodated within a 60 ns cycle time and this precludes a technique involving a read-modify-write cycle and RAM. The arithmetic is distributed instead throughout the memory. An array of bistables forming a set of asynchronous counting registers (64 off) within the columns, and shift registers (18 off) along the rows provides respectively for the accumulation and output of the histogram."

Bei Vergleich mit einer auf einem Speicherarray basierenden Architektur hat die Implementation des Histogrammteils als **Zählerarray** den Nachteil eines großen Flächenbedarfs und zusätzlich den Nachteil geringer Flexibilität, wenn die Zellen einfach gehalten werden sollen. Während zum Zeitpunkt der o.g. Realisierung nur NMOS-Technologie zur Realisierung eines derart komplexen Bausteins zur Verfügung stand, ist NMOS heute durch die CMOS-Technologie weitgehend verdrängt worden. In CMOS-Technologie realisierte Addierer und Speicher sind deutlich schneller als entsprechende in NMOS ausgeführte Komponenten.

Der Histogrammer soll mit einem sog. Standardzellen-Entwurfssytem implementiert werden. Ein Speicher der geplanten Größe kann mit einem Standardzellen-Entwurfssystem nur über vom Chip-Hersteller vordefinierte skalierbare Speicherblöcke realisiert werden. Für die Realisierung mit einem entsprechend großen Zählerarray müßte ein Full-Custom-Entwurf des Zählerarrays gemacht werden, da ein solches Zählerarray nicht als vordefinierter Block verfügbar ist.
Zur Speicherung des Histogramms soll ein Single-Port-RAM vorgesehen werden, da dieses das einzige in der geplanten Zieltechnologie beim MAZ zur Verfügung stehende RAM ist. Auch die kommerziellen Histogrammer [21] und [10] verwenden RAMs zur Histogrammspeicherung. Es ist jedoch nicht aus deren Datenblättern zu ersehen, welche Eigenschaften (wie Zugriffszeit und Portanzahl) diese RAMs aufweisen. Auf das RAM des Histogrammers soll für jedes Pixel ein **Read-Modify-Write**- (RMW-) Zugriff über eine zentrale ALU (Inkrementer bzw. Addierer) vorgenommen werden. Der Read-Modify-Write-Zugriff ist eine besondere Form des Zugriffs auf einen Speicher, bei dem jeweils eine Speicherstelle ausgelesen wird und anschließend ein neuer Wert in die gleiche Speicherstelle zurückgeschrieben wird. Dadurch,

daß zwischen Schreib- und Lesezugriff kein Adreßwechsel geschieht, kann ein solcher Zugriff meist schneller erfolgen als ein getrennter Schreib- und Lesezugriff.

Es ist vorauszusehen, daß die erreichbare Arbeitsfrequenz des Histogrammers beim geplanten Design mit einem normalen RAM durch die Geschwindigkeit des RAMs und den aufwendigen RMW-Zugriff stark eingeschränkt wird. Deshalb sollen hier Verbesserungsansätze untersucht werden.

Eine gängige Möglichkeit zur Geschwindigkeitssteigerung ist Pipelining. Der Aufbau eines RMW-Zugriffszyklus ermöglicht jedoch kein Pipelining (s. Absatz 4.3.2). Um Pipelining zu ermöglichen, müßte in jedem Zyklus von einer Adresse gelesen und auf eine unterschiedliche Adresse geschrieben werden können. Dazu müßte der RMW-Zugriff in einen getrennten Lese- und Schreibzugriff aufgeteilt werden. Eine solche Aufteilung wäre langsamer als ein Read-Modify-Write-Zugriff, da dann sowohl vor dem Lesezugriff als auch vor dem Schreibzugriff ein Adreßwechsel und eine Adreßdecodierung innerhalb des RAMs notwendig wird.
Ohne Geschwindigkeitseinbuße möglich ist diese Aufteilung nur mit einem Dual-Port-RAM, das über einen Schreibport und einen Leseport mit separaten Adreßeingängen und Steuersignalen verfügt (auch Two-Port-RAM genannt). Dieses belegt jedoch im Vergleich zu einem einfachen RAM eine größere Fläche auf dem Chip, da das Speicherarray über zwei Matrizen von Zeilen und Spalten adressierbar ist. Dazu werden zwei Adreßdekoder und auch aufwendigere Speicherzellen benötigt. In einer Standardzell-Speicherbibliothek [7] wurde sogar fast die doppelte Fläche verglichen mit einem Single-Port-RAM angegeben.

Mit einem **Dual-Port-RAM** wäre Pipelining im Histogrammteil möglich. Abbildung 4.3 zeigt das Prinzip eines solchen Pipelinings. Der Histogramminkrementer ist als Addierer ausgeführt, damit er wahlweise bei Erstellung von Kachelsummen als Addierer betrieben werden kann. Mit jeder Pixelankunft wird über den Leseport ein Pixel ausgelesen und erst mit dem folgenden Takt ins Eingangsregister des Addierers geladen. Während jedes Lesezugriffs bearbeitet der Addierer die zum vorigen Pixel gehörigen Daten. Währenddessen wird das vorherige Ergebnis des Addierers aus einem Halteregister über den Schreibport an die zugehörige Adresse geschrieben.
Diese Prinzipschaltung liefert jedoch falsche Ergebnisse, sobald eine Speicherstelle ausgelesen wird, für die sich noch ein Folgewert in der Pipeline befindet. Dann würde der bereits in der Pipeline befindliche Wert ohne Einfluß auf das Ergebnis bleiben. Für diesen Fall muß eine zusätzliche Logik vorgesehen werden, die anhand der Adressen in den Adreßpuffern überprüft, ob mehrfach nacheinander auf die selbe Speicherstelle zugegriffen wird. Stellt die Logik fest, daß eine Speicherstelle zweimal nacheinander gelesen wird, muß das Eingangsregister des Addierers beim zweiten Zugriff mit seinem aktuellen Ergebniswert anstatt mit dem ausgelesenen RAM-Datum geladen werden (sog. Data-Forwarding). Der Fall, daß eine Speicherstelle in einem Takt ausgelesen und beschrieben wird, wird von vielen Two-Port-RAMs bereits korrekt behandelt, indem der einzuschreibende Wert sofort auch an den Leseausgängen zur Verfügung steht.

Mit diesem dreistufigen Pipelining ist die Frequenz des Pixeltaktes dann nur noch über das langsamste Element in der Kette (Lesezugriff, Schreibzugriff oder Inkrementer) bestimmt.

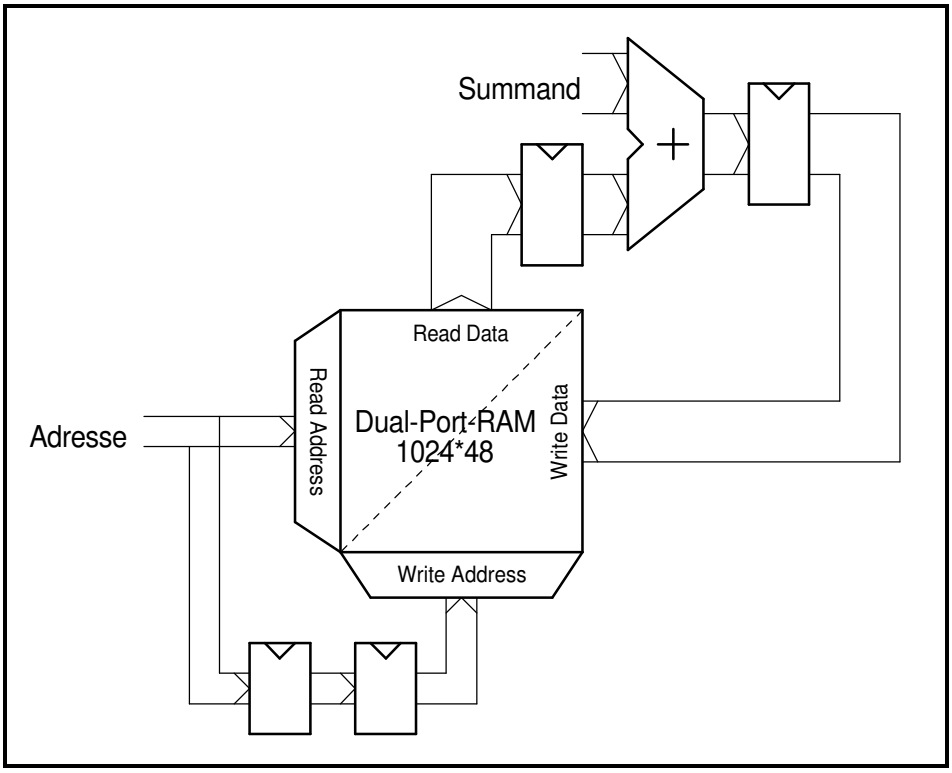

Abbildung 4.3: Pipelining mit Dual-Port-RAM (Prinzip)

Ebenfalls denkbar zur Geschwindigkeitssteigerung ist eine **doppelte Ausführung** des Histogrammarrays mit normalem RAM als Speicher. Beide Teile könnten parallel jedes zweite Pixel bearbeiten. Nach Aufzeichnung müßten beide Teilhistogramme addiert werden, um alle Pixel im Bild abzudecken. Da beide Speicherhälften über eine um nur ein Bit geringere Wortbreite verfügen dürften als ein einzelnes Array, wäre insgesamt fast der doppelte Speicherplatz notwendig.
Bei dieser Lösung wird die erreichbare Geschwindigkeit gegenüber der geplanten Ausführung verdoppelt, aber es ist auch nahezu der doppelte Flächenaufwand für Speicher und Inkrementer notwendig. Die Lösung mit einem Dual-Port-RAM verspricht bei ähnlichem Flächenaufwand höhere Geschwindigkeit.
Die Option der Aufteilung des Histogrammspeichers in zwei Teile wäre jedoch bei Realisierung des Speichers außerhalb des Chips in Betracht zu ziehen, da entsprechend große und schnelle Speicherchips sehr preiswert sind. Bei Realisierung des Histogrammers mit externem RAM entstehen Geschwindigkeitsverluste durch die Zeit, die die Signalübergabe zwischen Histogrammer und RAM benötigt. Auch ohne Verdopplung der ohnehin schon sehr großen Datenbusbreite von 48 Bit ergibt sich durch die Aufteilung des RAMs in zwei Teile die Möglichkeit, ein Pipelining einzuführen. Dieses könnte zweistufig sein, indem abwechselnd auf zwei Speicherbänke mit gemeinsamen Datenleitungen aber separaten Adreßleitungen zugegriffen wird (Interleaving). Dadurch stünde mehr Zeit für den Inkrementer und die Adreßübergabe an das RAM zur Verfügung.

Die geplante Realisierung des Histogrammers mit einem Single-Port-RAM erfordert den geringsten Hardwareaufwand, jedoch müssen die drei Operationen „Speicher auslesen“, „neuen Wert berechnen“ und „Ergebnis in den Speicher zurückschreiben“ für jedes Pixel seriell

erfolgen. Das Zwei- bis Dreifache der so erreichbaren Geschwindigkeit könnte bei Realisierung mit einem Dual-Port-RAM auf dem Chip erzielt werden.
Im folgenden sollen die für eine Realisierung mit einem Single-Port-RAM wesentlichen Komponenten detaillierter untersucht werden.

4.3.2 Read-Modify-Write-Zugriff auf den Histogrammspeicher

Die maximale Pixeltaktrate des Histogrammers wird bestimmt durch die Geschwindigkeit des Histogrammspeichers und des zentralen Inkrementers. Für jedes Pixel erfolgt ein Lesezugriff auf das RAM, anschließend werden die gelesenen Daten erhöht und zurückgeschrieben. Das RAM des Histogrammspeichers stellt ein aufwendiges Bauteil dar, dessen Ansteuerung in mehrere Phasen mit vom Chiphersteller vorgegebener minimaler Zeitdauer aufgeteilt werden muß.
Abbildung 4.4 zeigt den Read-Modify-Write-Zyklus bei einem RAM der Firma ES2 (CRAM ECPD 10 [7]), das ursprünglich als Histogrammspeicher verwendet werden sollte. Dieses RAM kann als skalierbare Makrozelle in den Standardzellentwurf integriert werden.

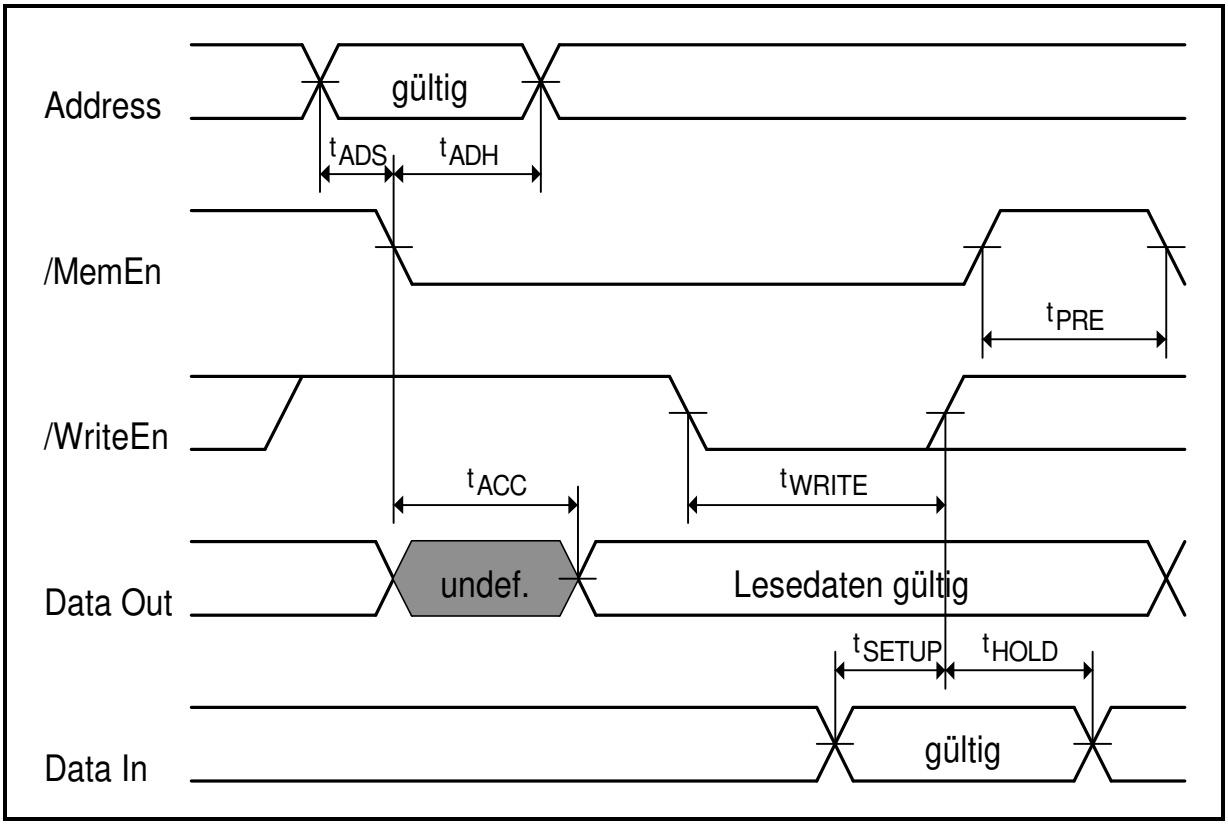

Abbildung 4.4: Read-Mofify-Write-Timing des ES2-RAMs